Conseil national de recherches Canada
National Research Council Canada

Division de génie électrique
Division of Electrical Engineering

ERB-998

Procédure d'ajustement et de calibration pour capteur tri-dimensionnel à balayage synchronisé

L. Cournoyer

Janvier 1987

Canada

87-04639/

Copyright 1987 par
Conseil national de recherches du Canada

Il est permis de citer de courts extraits et de reproduire des figures ou tableaux du présent rapport, à condition d'en identifier clairement la source.

Copyright 1987 by
National Research Council of Canada

Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.

Des exemplaires supplémentaires gratuits peuvent être obtenus à l'adresse suivante:

Bureau des publications, Pièce 301
Division de génie électrique
Conseil national de recherches du Canada
Ottawa (Ontario) Canada
K1A 0R8

Additional copies are available free of charge from:

Editorial Office, Room 301
Division of Electrical Engineering
National Research Council of Canada
Ottawa, Ontario, Canada
K1A 0R8

PROCÉDURE D'AJUSTEMENT ET DE CALIBRATION POUR CAPTEUR TRI-DIMENSIONNEL À BALAYAGE SYNCHRONISÉ

L. Cournoyer

ERB-998

CNRC Nº 27433

Janvier 1987

TABLE DES MATIÈRES

	Page
RÉSUMÉ	vii
ABSTRACT	viii
1 INTRODUCTION	1
2 AJUSTEMENT	3
2.1 Profondeur de champ et résolution	4
2.2 Mise au point optique	6
2.3 Champ de vision et résolution	8
2.4 Temps d'intégration et vitesse de balayage	9
3 ENREGISTREMENT	10
3.1 Images avec balayage dans deux axes	10
3.2 Images avec rotation de l'objet	12
3.3 Images de calibration	13
4 CALIBRATION	20
4.1 Format des images	20
4.2 Détermination des fonctions spline	20
4.3 Évaluation statistique du paramètre de correction H	23
5 CORRECTION	27
5.1 Correction des distorsions en X et en Z	27
5.2 Correction des distorsions en Y	28
5.3 Transformation des coordonnées cylindriques à cartésiennes	29
5.4 Interpolation sur une grille pour l'affichage graphique	30
5.5 Mesures	32
5.6 Affichage et mesures	34
REMERCIEMENT	41
RÉFÉRENCES	41
ANNEXE A Sources des programmes d'acquisition sur DPS-1	43
ANNEXE B Sources des programmes de calibration sur VAX-11/750	93
ANNEXE C Sources des programmes de correction sur VAX-11/750	117
ANNEXE D Sources des programmes de mesures sur VAX-11/750	145
ANNEXE E Sources des programmes d'affichage graphique sur VAX-11/750	155
ANNEXE F Sources des programmes de transformation et de lecture sur VAX-11/750	179

LISTE DES FIGURES

1 Schéma des composantes optiques du capteur tri-dimensionnel à balayage sur deux axes	3

LISTE DES FIGURES (suite)

		Page
2	Géométrie du capteur tri-dimensionnel	5
3	Représentation en 256 niveaux de gris de la distance D qui est codée sur 14 bits, d'un masque indien des Musées Nationaux du Canada	11
4	Image de la distance et des bits de saturation (15) et de validation (14)	11
5	Représentation en 256 niveaux de gris de la réflectivité I, codée sur 8 bits	12
6	Représentation en niveaux de gris d'une image prise avec rotation du masque sur 180°	14
7	Représentation en niveaux de gris d'une image de calibration en distance D	15
8	Schéma de la cible de calibration pour l'axe X	16
9	Représentation en tons de gris de l'image de réflectivité de la cible de calibration pour l'axe X	17
10	Représentation en niveaux de gris de la table de vérification de la fonction spline de calibration en Z	22
11	Représentation en niveaux de gris de la table de vérification de la fonction spline de calibration en X	24
12	Diagrammes logiques des programmes de calibration	26
13	Diagrammes logiques des programmes de correction, de transformation et de mesure	28
14	Représentation en niveaux de gris d'une image non corrigée de petits blocs usinés en aluminum sur un fond plat	31
15	Représentation en isométrie de l'image non corrigée de la figure 14	31
16	Représentation en isométrie de l'image de la figure 14, corrigée pour les distorsions en X, Y et Z	32
17	Affichage en niveaux de gris à gain fixe avec le bit de départ à 2, de l'image de la figure 3	35
18	Utilisation de curseurs pour cadrer une partie de l'image sur laquelle est appliquée le gain automatique	36
19	Affichage des mesures tri-dimensionnelles effectuées sur les données entre les deux curseurs graphiques, et agrandissement de la zone encadrée	37
20	Affichage en isométrie avec lignes horizontales	37
21	Affichage en isométrie avec lignes verticales	38
22	Affichage en isométrie avec les lignes dans les deux axes et sous-échantillonage pour diminuer la densité. Aussi, utilisation du décalage négatif	38
23	Affichage en isométrie de la portion de l'image cadrée de la figure 19 et une expansion dans les trois axes	39

LISTE DES TABLEAUX

Page

1 Fiche de calibration pour garder l;es paramètres de la caméra utile à la corrections 18

2 Tableau de mesures effectuées sur des points corrigés de l'image originale de la figure 3 33

RÉSUMÉ

Le capteur tri-dimensionnel à balayage synchronisé (1) fournit des données brutes qui sont proportionnelles à la distance (Z) qui sépare l'objet à mesurer d'une surface de référence. Le principe du balayage synchronisé et la non-linéarité des composantes optiques génèrent des distorsions qui ne peuvent pas être corrigées par des fonctions mathématiques simples.

Une calibration est donc nécessaire et consiste à prendre des données tri-dimensionnelles à des distances Z et à des angles de balayages connus de façon à générer deux tables de correction à partir des fonctions splines bi-dimensionnelles FZ(D, θ) et FX2(D, θ). Ces deux fonctions donnent respectivement les valeurs corrigées selon les axes Z et X et ce, pour les mêmes paramètres qui sont la sortie du capteur (D) et l'angle du balayage selon l'axe X (θ). Pour effectuer la correction du balayage selon l'axe Y (ϕ), on utilise une fonction trigonométrique simple.

Le présent rapport explique les différentes étapes qui conduisent à l'obtention d'une image tridimensionnelle corrigée sur laquelle des mesures peuvent être effectuées. On y explique l'ajustement de la caméra, l'enregistrement des données, la calibration et la correction des distortions ainsi que l'affichage et la prise de mesures sur les données. On y présente également le logiciel permettant ces différentes étapes et quelques résultats obtenus avec ces programmes.

ABSTRACT

The laser range finder based on synchronized scanning (1) provides raw data which are proportional to the distance (Z) between the object to be measured and the reference surface. The basic principle of synchronized scanning and the non-linearity of the optical components introduce some distortions which cannot be corrected by simple mathematical functions.

Therefore, a calibration based on tri-dimensional data from known distances (Z) and scanning angles (θ) is performed to create two correction tables interpolated by the two-dimensional spline functions FZ(D, θ) and FX2(D, θ). Those two functions give, respectively, the corrected values Z and X, for the same two parameters which are the output of the sensor (D) and the scanning angle (θ). The correction for the scan in the Y axis is done by a simple trigonometric transformation.

This report explains the different stages which lead to the acquisition and the correction of a tri-dimensional image on which measurements can be taken. It also describes the adjustments of the camera, the acquisition of data, the calibration and the correction of the distortions, and, finally, the display and measurements on the data. The software used is presented as well as the results obtained from the programs.

PROCÉDURE D'AJUSTEMENT ET DE CALIBRATION POUR CAPTEUR TRI-DIMENSIONNEL À BALAYAGE SYNCHRONISÉ

L. Cournoyer

1. INTRODUCTION

Le capteur tri-dimensionnel que nous utilisons pour digitaliser les objets fonctionne sur un principe de triangulation avec balayage synchronisé d'un faisceau laser. Ce principe a été développé au CNRC par Marc Rioux et a déjà été décrit en détail (1, 2).

Pour obtenir des images de haute qualité, le capteur nécessite un ajustement soigné au niveau de l'optique et de l'électronique. Le choix des composantes et leur ajustement dépend aussi du volume de mesure et de la résolution desirée. C'est cette procédure de mise au point qui est décrite dans le deuxième chapitre de ce rapport.

Il existe trois types de balayage possibles avec le capteur tri-dimensionnel (3). Le premier balayage est sur deux axes et la région balayée est rectangulaire. Le second type de balayage est effectué sur une ligne qui est normalement dans le plan de l'axe de rotation d'une table tournante sur laquelle est deposé l'objet à digitaliser. Enfin, il est possible d'effectuer un balayage point par point avec accès aléatoire. Il sera question seulement des deux premiers types dans ce rapport, le troisième n'ayant pas été expérimenté jusqu'à maintenant. Par contre, les images de calibration qui serviront à la correction des données sont assez différentes pour devoir s'y attarder. Le troisieme chapitre traite de ces trois types d'images et de la façon de les obtenir.

Le quatrième chapitre explique le processus de calibration de la caméra. Pour les distorsions en Z et en X, deux fonctions splines bi-dimensionnelles sont produites et la position des noeuds gardée en mémoire. Pour les images avec double balayage, il faut corriger pour la distorsion en Y, qui peut être représentée par une fonction trigonométrique simple. La calibration pour cet axe se résume à optimiser un paramètre de distance H dans la fonction. La caméra est calibrée lorsque tous les paramètres de ces fonctions sont déterminés.

Enfin ces fonctions servent à corriger les images qui proviennent du capteur et sur lesquelles peuvent être faites des mesures de distance. Elles peuvent aussi être affichées après avoir été interpolées adéquatement. C'est dans le cinquième chapitre que sont décrites ces dernières étapes nécessaires pour éliminer les distorsions sur les données originales et obtenir le maximum de précision du capteur tri-dimensionnel.

Par la description de ces différentes étapes, ce rapport est sous la forme d'une procédure qui devrait mener à un fonctionnement optimal du capteur. Bien que la description concerne seulement un des nombreux capteurs jusqu'à maintenant réalisés ou en voie de réalisation, cette procédure peut aider à mieux comprendre leur fonctionnement et leur mise au point et permettre d'en extraire les meilleures données tri-dimensionnelles.

2. AJUSTEMENT

Le prototype de capteur tri-dimensionnel sur lequel porte ce rapport combine deux galvanomètres placés à angle droit, l'un par rapport à l'autre, et qui produisent les deux balayages X et Y nécessaires à la formation d'une image tri-dimensionnelle (fig. 1). Le détecteur utilisé dans cette version est une barette CCD qui se compose de 256 éléments discrets. Ce capteur, conçu par le CNRC, a été fabriqué sous contrat par la compagnie Pocatec Ltée en 1983. Pour une description plus complète de la caméra, il faut se référer aux documents suivants :

- Documentation du contrat de Pocatec Ltée pour le projet de caméra 3D.
- Caméra construite par Pocatec Ltée - Manuel d'utilisation

L'explication des étapes d'ajustement suppose que la caméra est bien reliée à toutes les boîtes de contrôle qui s'y rattachent. Il est relativement facile d'y arriver en observant les connecteurs de la caméra

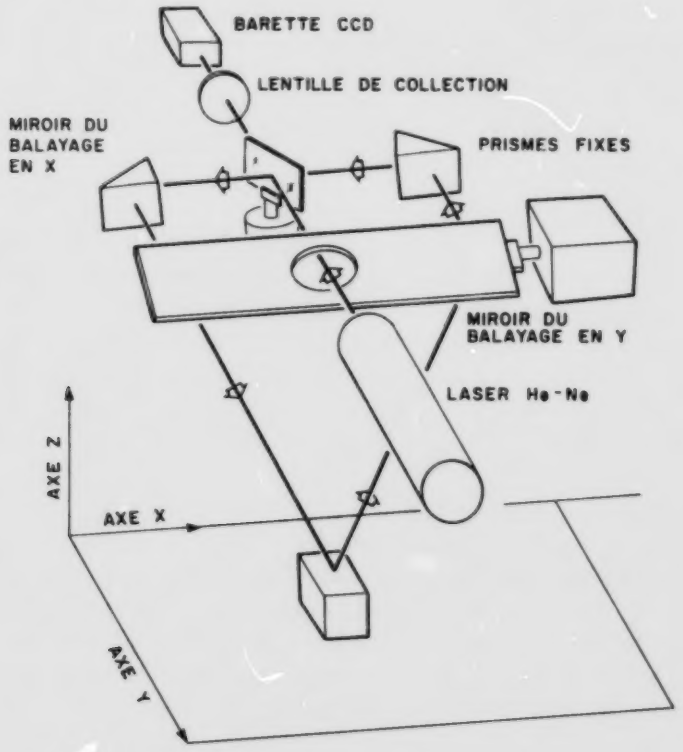


Figure 1. Schéma des composantes optiques du capteur tri-dimensionnel à balayage sur deux axes.

puisque aucune autre combinaison ne pourrait fonctionner. Pour plus de détails sur la connection des boîtiers entre eux, il faut se référer au manuel d'utilisation cité plus haut. Aussi on considère que les ajustements de l'électronique pour le contrôle et la détection ont été effectués comme il se doit. Il s'agit principalement du convertisseur vidéo analogue à numérique 8 bits de TRW, des comparateurs de détections de la position des galvanomètres et des interrupteurs du circuit de détection de maximum pour la barrette CCD. Ces ajustements sont également expliqués dans le manuel d'utilisation.

2.1. Profondeur de champ et résolution

La première chose à déterminer lorsque l'on veut digitaliser un objet est la profondeur de champ requise. Comme le nombre d'éléments de résolution à l'intérieur de la zone de mesure est constant pour un détecteur donné, la valeur d'un élément de résolution sera directement reliée à la profondeur de champ. Le rapport entre le déplacement dE sur le détecteur et la distance dZ correspondante dans le volume de mesure est le facteur de magnification M. Ce facteur dépend de la distance L qui sépare la lentille du plan de référence, de L_s qui est la distance entre le galvanomètre et le plan de référence, de F_0 qui est la distance lentille à détecteur et de D qui est la distance entre les deux prismes de côte (fig. 2) :

[1] $$M = dE/Z$$

[2] $$M = F_0 D/LL_s$$

Pour le prototype utilisé, la distance D est fixée à 8 cm. Si on veut changer la profondeur de champ et la résolution, il faut changer L ou F_0. La distance F_0 est proportionnelle à L et à la focale de la lentille de collection F selon la relation suivante :

[3] $$F_0 = FL/(L - F)$$

L'étape suivante consiste à trouver l'angle de la barrette pour que l'image du faisceau soit toujours au foyer sur la CCD. Cet angle est déterminé par le déplacement du foyer de l'image le long de l'axe optique et par le changement de position perpendiculaire à cet axe. Pour connaître le premier déplacement, il faut calculer la distance de focalisation de l'image F_0 lorsque le faisceau est à une distance donnée L. De la même façon, on trouve la distance F_1 lorsque le faisceau est à une distance (L - dZ) :

[4] $$F_1 = \frac{F(L - dZ)}{(L - dZ) - F}$$

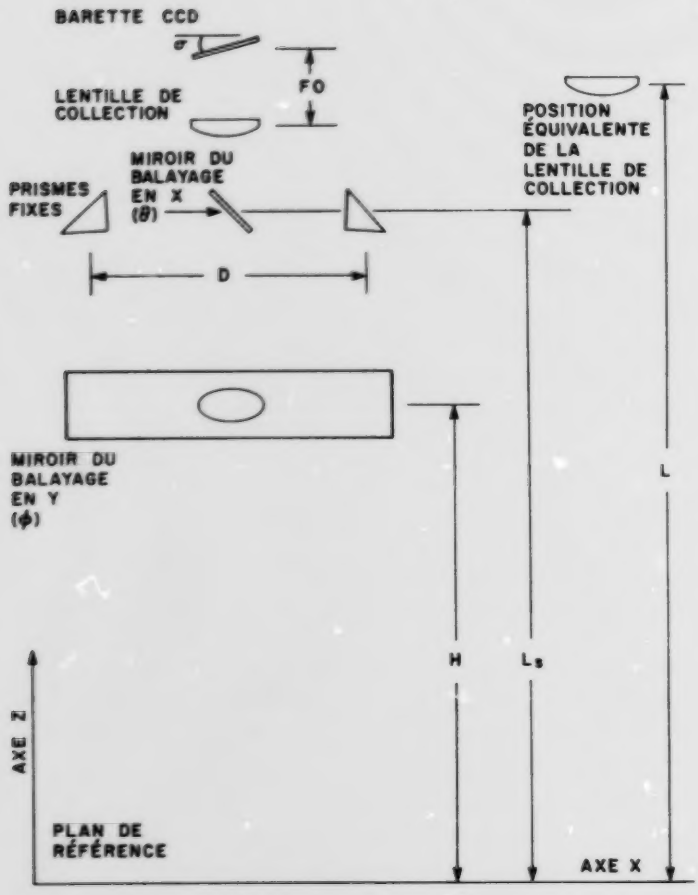


Figure 2. Géométrie du capteur tri-dimensionnel.

Le déplacement dE perpendiculaire à l'axe optique est proportionnel au déplacement du faisceau dZ et à la magnification du système M :

[5] $$dE = dZF_0D/LL_s$$

L'angle de la barrette correspond alors à :

[6] $$\sigma = \arctan [(F_1 - F_0)/dE]$$

Malgré la non-linéarité de la relation entre dZ et dE, on peut évaluer la profondeur de champ P et la résolution R du système. On remarque dans le calcul suivant que la dimension utilisée du côté du détecteur est la dimension apparente vue de la lentille à cause de l'angle de la barrette CCD :

$$[7] \qquad P = \frac{X\ (\cos \sigma)\ L^2}{F_0 D}$$

où X est la longueur de la barrette CCD.

Avec le circuit de détection de maximum développé par Blais (5), il est possible de détecter la position du maximum à une fraction d'élément de la barrette. La résolution est alors limitée par le système optique. Si on a un ajustement optique qui nous permet de résoudre 1/4 d'élément de barrette CCD et que celle-ci en possède 256, le nombre d'éléments de résolution est de 1000. La résolution R peut être approximée en divisant la profondeur de champ P par le nombre d'éléments de la barrette que l'on peut résoudre :

$$[8] \qquad R = P/N$$

où N est le nombre d'éléments de la barrette CCD que l'on peut résoudre avec le circuit de détection de maximum.

Il est possible d'obtenir la même profondeur de champ avec des lentilles de focales différentes. Par conséquent, il y a d'autres facteurs à considérer dans le choix de la lentille et de la distance L. Premièrement, l'intensité de lumière collectée est inversement proportionnelle au carré de la distance L. Ce qui veut dire que l'on a avantage à travailler le plus près possible surtout avec des objets de faible réflectivité. Par contre, les effets d'ombrages reliés à la triangulation augmentent lorsque la distance diminue et que l'angle de triangulation augmente. Il faut donc trouver le bon compromis entre ces deux contraintes pour chaque application.

2.2 Mise au point optique

Une fois que l'on a choisi la lentille et la distance de travail L, la caméra doit être ajustée pour cette distance. Cette mise au point de l'optique consiste à ajuster les miroirs ou prismes de côté, la dimension du faisceau incident, la focalisation de l'image sur la barrette CCD et la position de la barrette.

Pour l'ajustement des prismes de côté, les contrôleurs de galvanomètres doivent être allumés et à la position STOP et le GAIN à 10. On ajuste les contrôles OFFSET à la position médiane soit 5. En desserrant les vis qui maintiennent les prismes en place et à l'aide des vis d'ajustement de côté, on tourne le prisme de projection pour que le faisceau laser soit au centre de la plage à mesurer. Le prisme de collection doit avoir

le même angle que le premier. On peut optimiser la position de ce dernier pour avoir le maximum de lumière collectée par la lentille si on observe du côté du détecteur. Il ne faut pas resserrer tout de suite les vis qui retiennent les prismes de côté.

La prochaine étape consiste à focaliser le faisceau laser sur la surface. Ce faisceau doit être plus petit que la distance désirée entre deux échantillons. Si le faisceau est plus grand, il y aura un effet de filtrage sur les données, ce qui peut être désiré pour certaines applications où la surface est très rugueuse. On utilise du papier carton noir et des lunettes de protection pour le laser au rubis qui diminue la quantité de lumière et facilite l'observation. Un télescope de 4× est utilisé avec une lentille de 1 m pour focaliser le faisceau de 0,5 m jusqu'à plus de 2 m. Pour des distances de travail plus courtes, il faut rajouter une lentille d'environ 400 mm de focale. On ajuste alors le télescope pour obtenir le faisceau le plus petit possible pour l'ensemble de la distance de mesure.

On doit maintenant faire la mise au point de l'image sur la barrette CCD. L'observation peut être faite avec un oculaire placé temporairement au plan focal de la lentille de collection. Avant d'utiliser cet oculaire, l'intensité du laser doit être diminuée considérablement à l'aide de filtres de verres neutres, de telle sorte que l'observation ne soit pas inconfortable. En utilisant le mode INTERNE du contrôleur de galvanomètre du balayage en X et à l'aide de l'atténuateur GAIN, on balaye alors un objet qui couvre une bonne partie de la profondeur de champ désirée. L'image observée à l'oculaire devrait être une ligne droite. Si cette ligne n'est pas droite et qu'il y a un dédoublement qui forme un V, c'est que les prismes ne sont pas dans le même plan optique que le miroir de balayage. Il faut alors resserer les vis qui les maintiennent jusqu'à ce que la réflection se fasse sur une même droite, sans dédoublement.

La barrette peut alors être remise au plan focal de la lentille avec l'angle σ calculé plus haut. À l'aide d'un oscilloscope, on observe la sortie AMPLITUDE du détecteur de maximum en se synchronisant sur le signal THORIZ (B-B8), à l'arrière du contrôleur. La position de la barrette dans l'axe perpendiculaire à l'axe optique doit être celle où l'image du point qui se trouve au centre de la plage à mesurer se fasse au centre de la barrette. On ajuste maintenant la barrette en hauteur pour que l'image soit au foyer sur sa surface. L'image devrait couvrir environ 8 à 12 pixels de la barrette pour un fonctionnement optimal du circuit de détection de maximum qui utilise l'opérateur long [-1, -1, -1, -1, 1, 1, 1, 1]. Pour cette observation, les contrôleurs de galvanomètres sont placés à STOP. En se déplacant de chaque côté de la position médiane le signal AMPLITUDE devrait montrer un pic se déplaçant le long de l'axe X de l'oscilloscope

sans s'élargir. L'amplitude de ce pic peut toutefois diminuer avec la distance puisque l'intensité diminue.

Si le pic s'élargie lorsque l'on se déplace en distance, l'angle de la barrette doit être réajusté. Si l'amplitude diminue trop rapidement ou si elle diminue lorsque l'on se rapproche, c'est que le déplacement de l'image fait un angle avec la barrette et celle-ci doit être réalignée. Finalement, il faut vérifier que l'image demeure sur la barrette lorsque l'on balaye le faisceau en X et en Y. On peut se servir des ajustements OFFSET pour effectuer ces déplacements.

2.3 Champ de vision et résolution

L'optique de la caméra étant maintenant au point, on doit définir le champ de vision ou la plage angulaire à balayer. Les contrôleurs de galvanomètres doivent être à la position EXTERNAL et les ajustements de GAIN à 10 soit l'atténuation nulle. Les paramètres sont envoyés par un lien parallèle à partir du micro-ordinateur INTERSYSTEMS DPS-1. Le programme à utiliser est CAMPOC1 (annexe A). Comme le menu de ce programme l'indique, il y a quatre modes de balayages possibles. Les deux premiers produisent des balayages sur deux axes et la différence entre les deux modes est au niveau du balayage en Y. Le mode ESCALIER incrémente la position du miroir de balayage Y entre chaque ligne de balayage en X alors que le mode RAMPE déplace ce miroir de façon continue tout au long de l'image. Le troisième mode est le mode LIGNE où une seule ligne de balayage est effectuée entre les deux points limites spécifiés. Ce balayage utilise les deux galvanomètres. Enfin, le mode POINT prend une lecture de la caméra pour le point spécifié.

À partir d'un de ces modes, on envoie les paramètres du balayage à la caméra. Les paramètres peuvent être choisis en mode manuel ou chaque paramètre peut être modifié individuellement ou en mode interactif où le programme détermine lui-même les paramètres à partir de commandes interactives. C'est à ce moment que l'on détermine le champ de vision soit la différence entre la position initiale et la position finale. La résolution sera déterminée par l'incrément en X et en Y. C'est à cette étape aussi que l'on décide du nombre de points dans l'image. Il faut savoir que le nombre de points maximum pour une acquisition est de 64 Kpoints de 2 octets qui correspond à la mémoire maximum du DPS-1. Une fois les paramètres choisis, on peut les garder en mémoire avec la commande correspondante.

Pour vérifier le balayage, on peut le démarrer avec la commande DÉPART en ayant pris soin de mettre l'interrupteur OPERATION en mode INTERNE. La vitesse peut être maximum pour le temps de ces ajustements. La position finale du balayage peut être réajustée avec les potentiomètres OFFSET sur

les contrôleurs de galvanomètres. Une fois ces ajustements terminés, les potentiomètres des contrôleurs doivent être bloqués et ne plus être modifiés.

2.4 Temps d'intégration et vitesse de balayage

La vitesse du balayage du laser est reliée à la vitesse à laquelle la barrette CCD peut être lue. Or, la vitesse de lecture de la barrette dépend de la quantité de lumière reçue. Plus on a un objet de faible réflectivité ou un objet qui est loin de la caméra, plus il faut intégrer longtemps la lumière qui est collectée. Cela signifie que la vitesse de balayage maximum dépend de l'objet considéré et de sa position. On doit d'abord ajuster la vitesse d'intégration sans balayage par les micro-interrupteurs sur la carte F du détecteur de maximum, avec le faisceau laser sur l'objet. Le meilleur ajustement sera obtenue lorsque la LED verte de validation sera allumée pour l'ensemble du balayage et que la LED rouge de saturation ne sera pas allumée trop souvent. Il faut pouvoir trouver le bon ajustement pour garder l'intensité entre ces deux limites.

Pour trouver la vitesse de balayage appropriée au temps d'intégration, il faut faire une acquisition. On doit remettre l'interrupteur OPERATION en mode EXTERNE et appuyer sur la REMISE A ZÉRO. Le départ de l'acquisition doit se faire avec le programme CAMPOC1. Une ERREUR CCD survient lorsque le balayage du faisceau est trop rapide, c'est-à-dire lorsque le temps entre deux échantillons est plus court que le temps d'intégration. La vitesse doit alors être diminuée avec CAMPOC1. S'il y a une erreur, l'image n'est pas bonne et il faut recommencer l'acquisition. Pour ne pas avoir à attendre tout le temps un balayage d'une image pour changer la vitesse, on peut choisir les paramètres de balayage pour n'avoir que quelques lignes, le temps de l'ajustement. Comme l'erreur de CCD est occasionnée par un balayage trop rapide du faisceau pour la lecture de la CCD, il manquera des données à la fin du balayage et le micro-processeur sera en attente. Il faut donc repartir le balayage avec la commande manuelle DÉPART et lorsque l'on a une erreur d'ACQUISITION, c'est-à-dire quand l'image est completée, on effectue une REMISE A ZÉRO. À ce moment, on est prêt pour changer de vitesse de balayage et recommencer. S'il n'y a pas d'erreur pour tout un balayage de plusieurs lignes, la vitesse choisie est la bonne et on peut prendre une image complète.

3. ENREGISTREMENT

Avec le prototype de capteur tri-dimensionnel considéré, deux types de balayage sont possibles. Le balayage dans deux axes utilise les deux galvanomètres respectivement pour les balayages en X et en Y. Ce premier type produit des images à coordonnées cartésiennes sur un balayage rectangulaire. Le type cylindrique n'utilise qu'un galvanomètre qui balaye sur l'axe de rotation d'une table tournante sur laquelle est déposé l'objet. Le deuxième axe de l'image est l'angle de rotation de l'objet et le résultat est en coordonnées cylindriques. Enfin, les images de calibration sont des images qui servent à trouver les paramètres des fonctions de correction des distorsions géométriques.

3.1. Images avec balayage dans deux axes

Ce premier type de balayage a l'avantage de garder l'objet fixe et de ne pas ajouter d'erreur de vibration sur les données. Par contre, c'est une acquisition qui se fait d'un seul point de vue et qui est par conséquent relativement limitée si on veut digitaliser tout l'objet (p. ex. 360°). Il est toutefois possible d'intégrer plusieurs vues prisent à des angles différents et connus. Cette approche minimise les vibrations mécaniques tout en offrant le maximum de coordonnées de l'objet.

Ce balayage est obtenu avec le programme d'acquisition CAMPOC1. Une fois que tous les paramètres sont choisis et mis en mémoire, on peut débuter le balayage avec la commande ACQUISITION (CCD). Il existe deux types d'informations disponibles à l'acquisition, soit la distance (D) et la réflectivité (I). La distance est codée sur les 14 bits les moins significatifs du mot de 16 bits. La figure 3 représente l'affichage en tons de gris sur 8 bits des données de distance (D) (voir partie 5.6). Le bit 14 est le bit de validation, soit l'équivalent de la LED verte sur le contrôleur, et le bit 15 est le bit de saturation, donc ayant la même utilité que la LED rouge. On peut afficher l'image avec ces deux bits pour vérifier le nombre de pixels valides. La figure 4 est un exemple de cet affichage. Les régions les plus intenses de l'image sont les pixels où l'amplitude du signal fait saturer le convertisseur, soit lorsque le bit 15 est actif. Les régions où l'amplitude est suffisante, donc où les pixels sont valides, et le bit 14 est actif sont représentées par la teinte grise sur la figure 4. Le reste de l'image soit la région noire représente la région ou l'intensité n'est pas suffisante. Les images doivent contenir le plus d'informations valides possibles, soit la teinte grise.

La réflectivité (I) est codée sur 8 bits seulement et il n'y a pas de bit de validation comme précédemment. L'affichage peut se faire directement en tons de gris puisque le nombre de bits d'affichage est aussi de 8 bits. Le BIT DE DÉPART pour l'affichage de la figure 5 est le bit 1. Le

Figure 3. Représentation en 256 niveaux de gris de la distance D qui est codée sur 14 bits, d'un masque indien des Musées Nationaux du Canada.

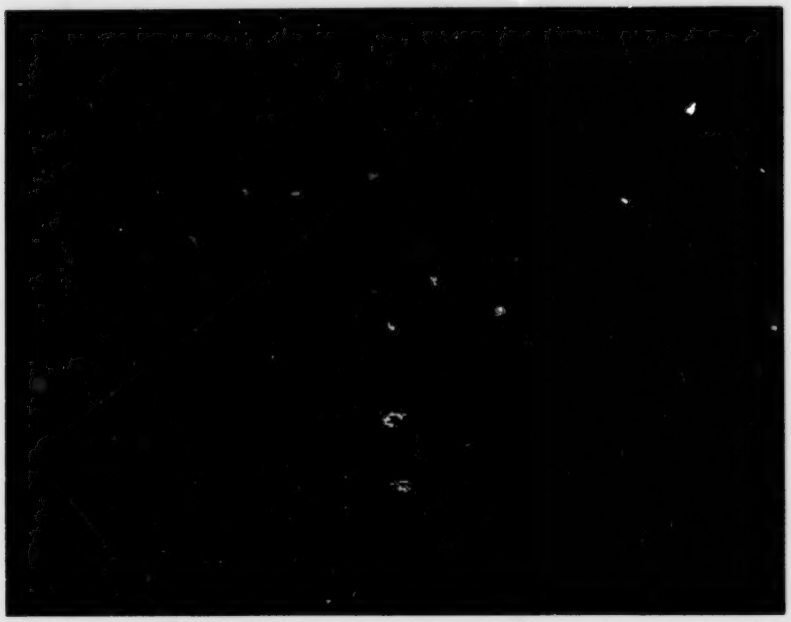


Figure 4. Image de la distance et des bits de saturation (15) et de validation (14).

choix du genre de données se fait à partir des micro-interrupteurs de la carte B. Le code 01 signifie la distance et le code 10 la réflectivité. Il est aussi possible de faire une acquisition double soit les deux genres de données en alternance (code 11). Dans le programme CAMPOC1 cette acquisition est specifiée par l'extension (UDT), soit 4 octets par échantillon.

Les images sont ensuite envoyées à l'ordinateur VAX-11/750 par un lien parallèle de 16 bits. Avec le programme CAMPOC1, il faut choisir l'option I pour le transfert de données. Dans le VAX-11/750, le programme à utiliser est XFER suivit du nom du fichier que l'on veut écrire. Tous les fichiers créés dans le VAX-11/750 avec XFER sont de longueur de registre fixe à 1024 octets. Pour les convertir en longueur de registre variable, on utilise TRANSVAR, auquel il faut fournir les dimensions de l'image. Cette opération doit être exécutée pour tous les fichiers qui sont transférés avec ce programme. Les images peuvent être affichées sur le TRAPIX avec le programme AFF dans le VAX-11/750 (voir annexe E).

3.2 Images avec rotation de l'objet

Le programme servant à faire des acquisitions avec rotation de la table tournante est DIGIS1. Ce programme utilise le mode ESCALIER de la

Figure 5. Représentation en 256 niveaux de gris de la réflectivité I, codée sur 8 bits.

caméra et incrémente la table tournante à chaque balayage en X. Il faut s'assurer que le balayage se fasse sur l'axe de rotation de la table, et ce même lorsqu'il y a déplacement de la table dans l'axe Z, soit lorsque la table se rapproche ou s'éloigne de la caméra. Cet ajustement est très important puisque la calibration et la transformation des données cylindriques en coordonnées cartésiennes ne tient pas compte de cette erreur. Il est donc nécessaire de la minimiser. On peut utiliser l'équerre sur l'axe de rotation pour une bonne mesure.

L'ajustement de l'angle de la table tournante pour chaque incrément doit être déterminé manuellement sur le contrôleur de la table de rotation. Une unité sur le contrôleur correspond à 1/100 de degré de rotation. Le nombre d'incréments ou de lignes pour une image est à définir dans les paramètres du programme DIGIS1. De la même façon que précédemment, le nombre total de points de 16 bits ne doit pas dépasser 64 Kpoints par image. Il est possible de joindre plusieurs images prisent côte à côte une fois dans le VAX-11/750 avec le programme JONCTION. Pour plus de flexibilité, on peut changer la position de la table tournante dans l'axe des Z entre deux images. Mais il est très important de garder en note la distance entre les deux images, ainsi que l'angle entre deux lignes de balayage.

Avec le programme DIGIS1, il faut choisir les paramètres du balayage en X. La position du faisceau en Y est determinée par les paramètres de PHY INITIAL et PHY FINAL qui doivent être tous deux égaux au nombre qui positionnera le faisceau sur l'axe de rotation de la table. Il faut définir les ANGLE INITIAL et ANGLE FINAL de la rotation de la table ainsi que l'INCRÉMENT. On choisit la vitesse de balayage de la même façon que pour le mode rectangulaire. Le contrôleur qui est incrémenté est le #1 sur le triple contrôleur DAEDAL. Aussi, il faut savoir que le programme fait toujours l'acquisition d'une ligne pour l'angle initial et l'angle final (nombre de positions = (angle final - angle initial) + 1). Il faut déterminer le temps d'attente minimum entre deux balayages qui permettra aux vibrations de s'atténuer suffisamment. La figure 6 représente un objet pris avec rotation de la table tournante, affiché en niveaux de gris. L'angle α entre chaque ligne est de 0,9°.

3.3 Images de calibration

Pour corriger les données en provenance du capteur tri-dimensionnel, il faut effectuer une calibration. Une partie de cette calibration consiste à prendre une image d'un plan à différents endroits dans l'espace le long de l'axe Z pour connaître la distorsion du capteur en fonction de la distance et de l'angle de balayage. Il y a deux images nécessaires pour faire la correction linéaire en X et Z. Une autre image doit être prise

Figure 6. Représentation en niveaux de gris d'une image prise avec rotation du masque sur 180°.

soit pour la correction dans le sens du balayage Y, s'il s'agit d'une image rectangulaire, ou pour la correction par rapport à l'axe de rotation s'il s'agit d'une image cylindrique.

Les deux images servant à la calibration des axes X et Z sont prisent avec le programme DIGIS1. Ce sont des images avec un seul balayage soit celui de l'axe X. L'autre galvanomètre est fixé à peu près au centre de la plage à balayer ou sur l'axe de rotation s'il s'agit d'une image prise en rotation. L'amplitude du balayage en X doit être la même que celle des images que l'on veut prendre. La position des OFFSET ne doit pas être changée pendant la prise d'image et la calibration.

Le programme DIGIS1 permet de prendre plusieurs lignes à la même position en spécifiant le nombre avec le paramètre NOMBRE DE PASSE D'INTÉGRATION. Bien que le programme soit prévu pour l'acquisition avec la table tournante, on peut l'utiliser si on change les câbles pour contrôler une table de translation plutôt qu'une table de rotation. Le contrôleur utilisé est le #1 sur le triple contrôleur DAEDAL. On peut spécifier le nombre de position que l'on veut enregistrer avec les paramètres ANGLE INITIAL, ANGLE FINAL et INCRÉMENT DE LA TABLE TOURNANTE. Il ne faut pas oublier que le programme fait toujours l'acquisition d'une ligne pour l'angle initial et l'angle final. On peut fixer le balayage de l'axe Y en

spécifiant l'angle ϕ INITIAL et ϕ FINAL avec le nombre qui positionnera le faisceau à l'endroit désiré pour cet axe soit le milieu de la plage ou l'axe de rotation de la table tournante selon le type de balayage qui sera utilisé.

On ajuste manuellement la première position du plan en Z le plus loin possible de la caméra tout en s'assurant que l'image du faisceau ne sort pas de la CCD. Pour le vérifier, il faut balayer en X avec la commande DÉPART dans le mode OPÉRATION INTERNE. De la même façon, on trouve la position la plus proche de la caméra pour laquelle le signal est bon pour tout le balayage. En connaissant ces deux positions extrêmes, on détermine le nombre de positions différentes et la distance entre chacune. Le nombre de position doit être d'environ 10. La distance entre chaque position doit être déterminée par les interrupteurs du contrôleur DAEDAL. La précision de la table est de 0,0005 po. On remet le plan à la position la plus loin de la caméra et on met l'interrupteur START du contrôleur DAEDAL dans la position EXTERNAL. On remet à EXTERNE l'interrupteur OPÉRATION de la caméra avant de revenir au programme DIGIS1.

La caméra est maintenant prête pour l'acquisition. La commande de DÉPART du programme devrait débuter le balayage. La première image est du type tri-dimensionnel et l'objet est un plan d'une reflectivité suffisante

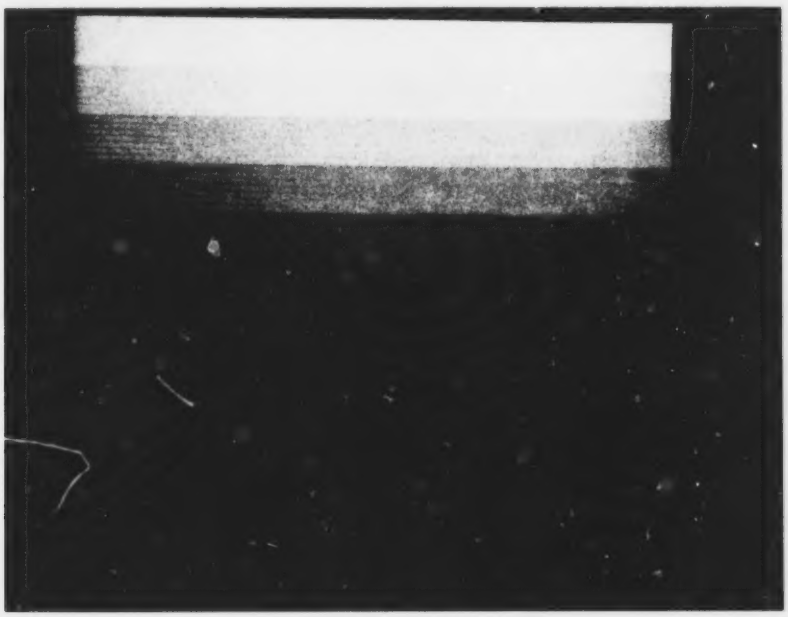
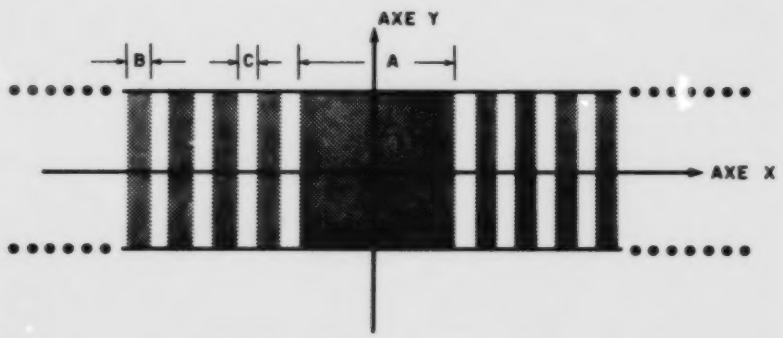


Figure 7. Représentation en niveaux de gris d'une image de calibration en distance D.

pour avoir un bon signal sur tout le balayage. La figure 7 montre une vue en tons de gris de cette image de calibration. Chaque transition de tons de gris correspond à une nouvelle position en distance Z.

La seconde image sert à calibrer l'axe X. L'image doit être prise aux mêmes positions que la précédente mais elle est du type réflectivité plutôt que tri-dimensionnelle. Aussi, la cible est différente de la première puisqu'elle est un plan noir avec des franges blanches de dimensions fixes à une distance fixe les unes des autres. Les deux franges du centre sont à une distance un peu plus grande pour conserver une référence d'une position à l'autre (fig. 8). Entre les deux images, il faut remettre le plan à la position la plus loin de la caméra. La figure 9 est une représentation en tons de gris de l'intensité. On distingue nettement les franges pour chaque position en distance Z. La cible doit être choisie en fonction de la résolution en X. Il existe quatre cibles différentes dont la dimension des franges varient respectivement de 2,00, 0,75, 0,30 et 0,10 mm. Les distances centre à centre entre les franges sont de 10,00, 3,75, 1,50 et 0,50 mm. Les distances entre les franges centrales sont de 15,0, 6,0, 2,5 et 0,75 mm.

Pour corriger les images prisent avec deux balayages, soit X et Y, on peut corriger les distorsions sur l'axe Y par une fonction trigonométrique simple. Cependant, cette fonction utilise des paramètres qu'il faut mesurer le plus précisément possible. Ces paramètres sont l'angle de départ du balayage ainsi que l'incrément angulaire. L'angle de départ est pris



A LA DISTANCE ENTRE LES FRANGES CENTRALES

B LA DISTANCE ENTRE LES FRANGES

C LARGEUR DES FRANGES

Figure 8. Schéma de la cible de calibration pour l'axe X.

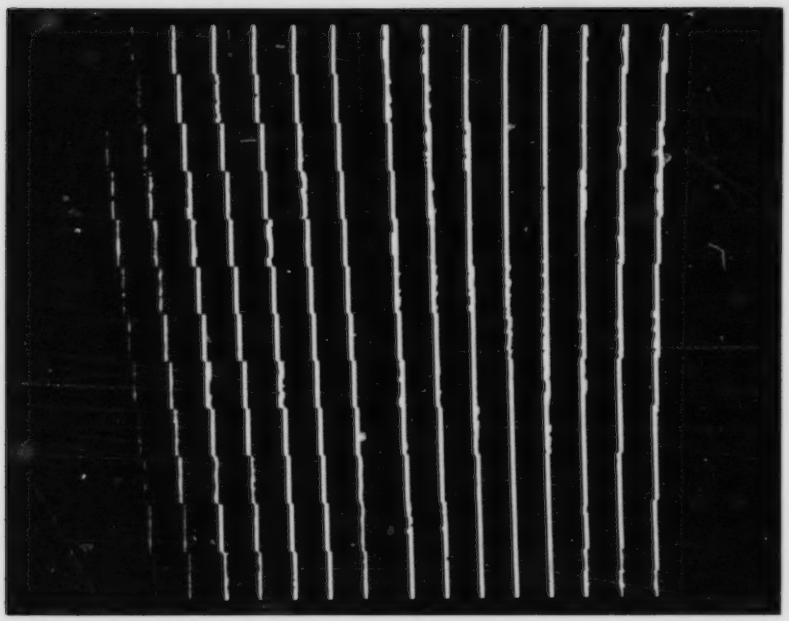


Figure 9. Représentation en tons de gris de l'image de réflectivité de la cible de calibration pour l'axe X.

par rapport à l'angle zéro. L'angle zéro est la position pour laquelle le faisceau ne se déplace pas en Y pour un déplacement en Z. Pour mesurer ces angles, on se sert des programmes CAMPOC1 ou DIGIS1 avec lesquels il est possible de déplacer le faisceau dans l'axe Y seulement, de mesurer le déplacement et par conséquent l'angle de balayage. Ensuite, on prend une troisième image de calibration qui est un plan le plus perpendiculaire possible à l'axe optique et à une position connue par rapport au deux premières images de calibration.

Par contre, si on veut faire la correction des images de rotation, la troisième image de calibration est une image de l'axe de rotation. Cette image consiste en plusieurs lignes d'un plan qui est placé précisément sur le centre de rotation. Ces lignes sont moyennées avec MOYLIGNE et la ligne résultante permet d'évaluer la position moyenne de l'axe de rotation que l'on soustrait des données lors de la correction. Cette image peut être prise avec CAMPOC1 ou DIGIS1 pourvu que la position Y du faisceau soit fixée sur l'axe de rotation de la table tournante.

Les paramètres qu'il faut conserver lors de l'acquisition peuvent être rassemblés dans une fiche de calibration (tableau 1).

Tableau 1. Fiche de calibration pour garder les paramètres de la caméra utile à la corrections.

Date : _____

FICHE DE CALIBRATION CAMERA 3D

Sujet : _____

Parametres du balayage en X (teta) :

OFFSET	INITIAL	FINAL	INCREMENT
_____	_____	_____	_____

Parametres du balayage en Y (phy) :

OFFSET	INITIAL	FINAL	INCREMENT	ROTATION
_____	_____	_____	_____	_____

Angles (double balayage) :

INITIAL	INCREMENT
_____	_____

Angles (rotation) :

INITIAL	INCREMENT
_____	_____

Distances :

FOCALE	GALVO Y A POSITION #1	LENTILLE A DETECTEUR	UNITE
_____	_____	_____	_____

Images :

PARTITION : _____

NOM	DIMENSIONS	DESCRIPTION

Programmes d'acquisition (annexe A) :

- CAMPOC1 : Programme général d'acquisition de données.
- DIGIS1 : Programme d'acquisition sur une ligne avec incrémentation d'une table de positionnement de translation ou de rotation.
- RAMPE : Sous-programme d'acquisition de données avec balayage en Y continue.
- ESCAL0 : Sous-programme d'acquisition de données avec balayage en Y par incréments.
- LIGNE : Sous-programme d'acquisition de données sur une ligne mais avec double balayage.
- POINT : Sous-programme d'acquisition de données pour chaque point individuellement.
- ACQUI1 : Sous-programme d'acquisition d'un solide de révolution utilisant le mode escalier de la caméra.
- SETXY : Sous-programme d'initialisation des paramètres de la caméra en mode interactif.
- DEFACQ : Sous-programme de définition des paramètres de l'acquisition d'un solide de révolution.
- PARAL : Sous-programme de transfert de données sur un bus de 16 bits parallèle.

4. CALIBRATION

4.1. Format des images

Jusqu'ici, les données proviennent toutes du programme de transfert XFER. Que ces données soient tri-dimensionnelles ou de réflectivité, l'information est contenue dans un mot de 16 bits soit un entier de 2 octets. À partir de cette étape, les images créées sont des modifications des images brutes et pour éviter les erreurs de troncations, l'information est codée en réel de 4 octets. Le format commun à toutes les images produites par les programmes de calibration et de correction est celui de DPICT produit par DATAPLOTTING SERVICES INC., qui est utilisé dans un ensemble de programmes d'affichages graphiques. Dans ce format, le premier registre des images contient l'information sur leurs dimensions et leurs facteurs d'échelle. Les programmes de calibration et de correction ajoutent dans ce registre de l'information que DPICT ne prévoit pas, soit les facteurs d'échelle et les valeurs minimums et maximums pour X et Y. Mais ces fichiers restent compatibles avec les programmes d'affichage de DPICT. Un autre programme d'affichage sur moniteur vidéo a été developpé et fonctionne avec le processeur d'images TRAPIX de RECOGNITION CONCEPTS INC. Ce programme qui utilise le format DPICT, est AFFI1 qui produit un affichage en isométrie et en niveau de gris et permet d'effectuer des mesures (annexe E).

Pour conserver les données telles qu'elles ont été corrigées par les fonctions SPLINE, deux autres formats sont créés. Le premier utilise le premier registre pour y mettre l'information concernant les dimensions et le facteur d'échelle en Z tout comme le format DPICT. Les trois registres suivants sont les tableaux X, Y et Z contenant les coordonnées des points corrigés. Si les points ne sont pas valides, ces tableaux contiennent la valeur zéro. Ce fichier est utilisé pour un éditeur de solide développé dans la section d'Automatique du CNRC (3).

Finalement, on peut créer une image de format DPICT pour chacun de ces trois tableaux pour pouvoir visualiser les coordonnées individuellement sur l'écran graphique. Ces images peuvent être créées seulement au besoin puisqu'elles prennent beaucoup d'espace en mémoire.

4.2 Détermination des fonctions spline

Comme il a déjà été expliqué dans la deuxième partie, la première étape de la calibration consiste à garder en mémoire une représentation de la distorsion du capteur. Cette distorsion peut être mémorisée à l'aide de tables de calibration qui peuvent être plus ou moins volumineuses (4). L'approche suivante est une représentation sous la forme d'une fonction

SPLINE bi-dimensionnelle (6). On se sert d'images d'un plan pris à différentes positions connues dans l'espace de mesure pour créer une fonction SPLINE qui représente la sortie du capteur et pour laquelle on ne garde en mémoire que les paramètres de base. Pour corriger une image, on se réfère à cette fonction pour retrouver les coordonnées exactes dans l'espace. Cette approche a l'avantage de nécessiter très peu d'espace mémoire comparativement à garder une table de calibration. L'inconvénient est le temps de calcul requis pour approximer chaque point que l'on veut corriger avec la fonction SPLINE. Par contre, ce calcul n'introduit pas d'erreur de troncation comme c'est le cas avec les tables.

4.2.1. Calibration pour la distorsion en Z

Le programme qui produit les paramètres de la fonction SPLINE pour les distorsions de l'axe Z se nomme ZCALI. Il utilise le premier fichier de calibration soit un plan que l'on déplace à plusieurs positions d'une distance fixe. Aussi on peut intégrer plusieurs lignes pour chaque position. Le sous-programme SURFAC1 est utilisé pour calculer les paramètres de la fonction. Ces paramètres qui sont les noeuds de la fonction SPLINE, sont gardés dans un fichier dont le nom est demandé à l'utilisateur. Il est préférable que tous les noms créés pour une série d'image aient un caractère commun pour nous permettre de les associer ensemble (p. ex. : masque.Z, masque.X sont les fichiers de calibration pour les images du masque).

Pour vérifier la validité des fonctions SPLINE et s'assurer qu'il n'y ait pas d'erreur majeure, il est possible de générer une table à partir des paramètres de la fonction. Cette table n'est pas nécessaire à la calibration et elle peut être effacée après celle-ci. Elle est produite par le programme GENTAB et est de format DPICT. La dimension de cette table est déterminée par le nombre de lignes que l'on spécifie dans le programme. Elle peut être affichée avec AFFI1 ou DPICT (fig. 10). Le programme ZCALI fournit au sous-programme SURFAC1 trois tableaux de coordonnées XX, YY et ZZ avec lesquels le sous-programme trouve les paramètres de la fonction SPLINE qui les représentent. Il s'agit par la suite de donner une coordonnée XX et YY pour pouvoir interpoler la valeur ZZ correspondante (6). Pour le programme ZCALI, XX représente la position du pixel dans l'image, YY la donnée tri-dimensionnelle venant du capteur, et ZZ la position connue du plan de référence. Le programme SURFAC1 effectue un certain filtrage sur les données en les suivants plus ou moins rigidement. Cette rigidité est contrôlée par le facteur d'amortissement S qui peut être évalué comme le produit de l'écart type du bruit sur les données par le nombre de points utilisés. Cet écart type peut être trouvé par le sous-programme ZFACT qui lit le fichier de calibration et détermine le niveau de bruit à chaque position angulaire et pour chaque position en Z. L'écart type est calculé à

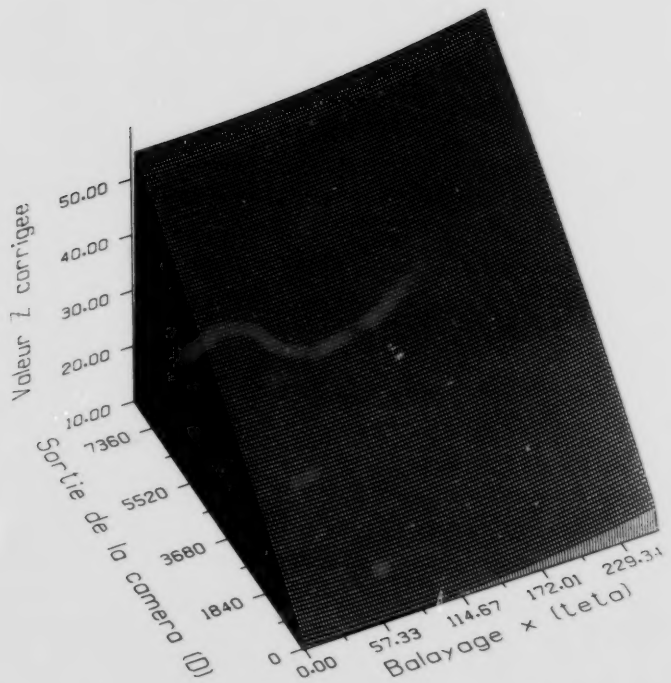


Figure 10. Représentation en niveaux de gris de la table de vérification de la fonction spline de calibration en Z.

partir de ce niveau de bruit. Il est aussi possible d'entrer un facteur d'amortissement différent du facteur calculé pour plus ou moins accentuer l'effet de filtrage (annexe B). On a donc une fonction SPLINE qui peut approximer une valeur de distance Z pour deux valeurs d'entrée qui sont la position angulaire (θ) du faisceau et la lecture de la barrette CCD (D), soit une fonction $FZ(\theta, D) = Z'$.

4.2.2. Calibration pour la distorsion en X

La calibration du capteur pour l'axe X se fait d'une façon similaire à celle de l'axe Z. Cette fois, il faut connaître la position exacte en X du faisceau sur le plan de référence ainsi que sa position en Z. On utilise un plan de référence qui est une série de franges blanches sur un fond noir ou l'inverse et qui sont placées perpendiculaires au balayage. On utilise l'image de réflectivité pour y distinguer les pics qui correspondent à chacune des franges. Ces franges sont à une distance fixe et connue les unes des autres exceptées les deux du centre qui sont un peu plus distancées que les autres. Le programme XCALI1 lit cette image de calibration et détecte la position des maximums de chaque pic dans l'axe des X en fractions de pixel. La position des maximums est trouvée à l'aide du même algorithme

que la position du faisceau sur la barrette CCD (5). On fait le balayage de ce plan de référence aux mêmes positions en distance dans l'axe des Z que la première image de calibration. Pour chaque position, une liste des positions des pics détectés est affichée. Ces listes permettent de distinguer les pics qui n'ont pas pu être détectés. On peut les comparer à l'image d'intensité affichée plus haut (fig. 9).

De la même façon que pour le programme ZCALI, on se sert de SURFAC1 pour créer une fonction SPLINE intermédiaire. Les tableaux d'entrée cette fois sont : pour XX la position des franges en fractions de pixel, pour YY la position du plan dans l'axe Z en unités et pour ZZ la position des franges en unités. La fonction peut être représentée par FX1(θ, Z) = X. La position médiane entre les deux franges du centre est détectée à chaque position en Z et sert de référence dans l'axe X pour toutes les positions. La position du zéro est définie à l'extérieur de l'espace de mesure pour les deux axes X et Z.

Finalement, une troisième fonction SPLINE similaire à celle créée par ZCALI, fournit une valeur X pour les deux mêmes valeurs d'entrée soit la position angulaire (θ) du faisceau et la lecture de la barrette (D) et est représentée par FX2(θ, D) = X. Cette fonction est produite à partir de la fonction intermédiaire créée par XCALI1 et la première image de calibration. Les tableaux d'entrée sont, pour XX la position du faisceau en pixel, pour YY la lecture de la première image de calibration pour chaque position du plan dans l'axe Z, et pour ZZ le résultat de la fonction SPLINE intermédiaire pour le pixel considéré et la position en Z considérée. Les paramètres de la fonction SPLINE résultante sont sauvés dans un fichier sous le nom que l'utilisateur lui donnera dont l'extension est PAR. Il est aussi possible de créer une table de vérification pour cette fonction SPLINE avec le programme GENTAB (fig. 11).

4.3. Évaluation statistique du paramètre de correction H

Jusqu'alors, la calibration n'est faite que pour le balayage dans l'axe X. Pour les images générées à partir de deux balayages, une calibration supplémentaire est nécessaire. Comme le deuxième balayage est généré par un miroir tournant sur un axe, la distorsion peut être corrigée par calcul à partir d'une fonction trigonométrique simple. La correction consiste à trouver la coordonnée Y et à corriger celle de Z déjà trouvée avec la fonction SPLINE précédente. Les fonctions trigonométriques sont les suivantes :

$$[9] \qquad Y = (H - Z') \sin(\phi)$$

$$[10] \qquad Z = H - (H - Z') \cos(\phi)$$

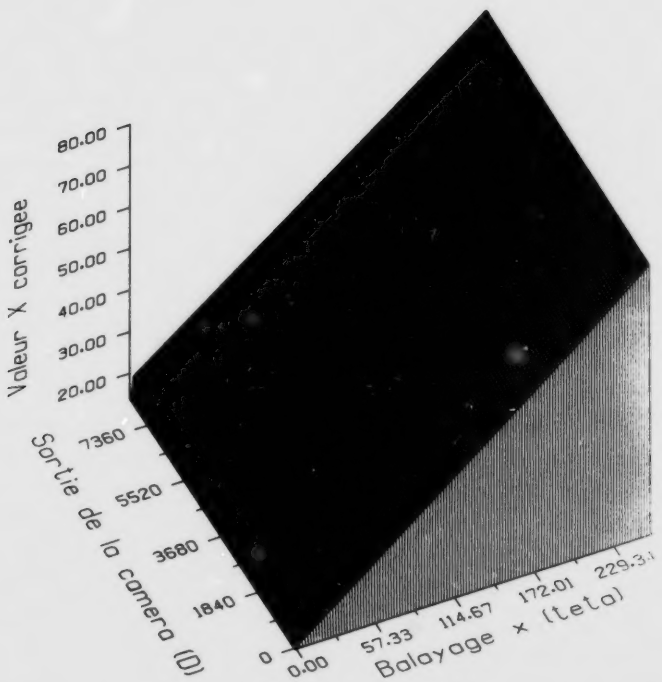


Figure 11. Représentation en niveaux de gris de la table de vérification de la fonction spline de calibration en X.

où

Z' représente la valeur de la distance corrigée pour le premier balayage en X par la fonction SPLINE, FZ(θ, D),

H représente la distance entre l'axe de rotation et le plan de référence le plus loin;

ϕ représente l'angle de projection du faisceau par rapport à l'axe optique pour ce balayage.

Pour que la correction soit valide, il faut que les paramètres H et ϕ soient les plus précis possible. Il est difficile en pratique de mesurer la valeur de H. La calibration pour ce balayage consiste à trouver une valeur optimisée de H qui donnera la meilleure correction pour l'ensemble d'une image de référence. On se sert alors de la troisième image de calibration qui est un plan de référence balayé avec la même amplitude que les images que l'on veut corriger et pour lequel on connaît la position exacte par rapport aux premières images de calibration. Aussi, on aura pris soin de mesurer l'angle de départ du balayage en Y ainsi que l'incrément

angulaire (section 3.3). Aussi, il faut s'assurer de bien mettre le plan perpendiculaire au balayage.

Le programme YCALI permet de choisir une valeur optimisée de H. Il faut fournir au programme les valeurs de départ pour le balayage et la position en Z du plan pour cette image. Le programme fait le calcul inverse et trouve la valeur de H pour tous les points de l'image. Aussi, on peut spécifier une marge angulaire autour de l'angle zéro pour laquelle le calcul de H ne doit pas être considéré à cause du manque de précision. Le programme présente sur l'écran un histogramme des valeurs de H calculées. Les bornes de l'affichage de l'histogramme ainsi que la largeur des intervalles sont des paramètres demandés à l'utilisateur. La valeur moyenne ponderée de H est calculée seulement pour les données faisant partie de l'histogramme qui est affiché.

Si l'histogramme ne contient qu'un pic avec une distribution presque gaussienne, c'est que les paramètres sont bien optimisés. Par contre, s'il y a deux pics ou plus, un des paramètres de départ est erroné et il faut recommencer avec d'autres valeurs. Il faut se servir des bornes pour limiter le calcul de la valeur moyenne pondérée aux valeurs les plus probables. Aussi, en diminuant l'incrément de l'histogramme, la moyenne est mieux pondérée pour chaque intervalle. Les paramètres choisis sont écrits dans un fichier dont le nom est demandé à l'utilisateur et qui servira lors de la correction. La figure 12 représente sous forme de diagrammes logiques tous les programmes servant à la calibration de la caméra, ainsi que des fichiers qui sont créés.

Programmes de calibration (annexe B) :

- ZCALI : Programme de détermination d'une fonction spline bidimensionnelle représentant la distorsion du capteur dans l'axe Z sous la forme $FZ(\theta, D) = Z$.
- XCALI1 : Programme de détermination d'une fonction spline intermédiaire représentant la distorsion en X, soit $FX1(\theta, Z) = X$.
- XCALI2 : Programme de détermination d'une fonction spline représentant la distorsion en X sous la forme $FX2(\theta, D) = X$.
- GENTAB : Programme de génération de table de vérification des fonctions de calibration FZ et FX2.
- ZFACT : Sous-programme de calcul de l'erreur quadratique moyen à partir du fichier de calibration en Z.
- XFACT : Sous-programme de calcul de l'erreur quadratique moyen à partir du fichier de calibration en X.
- DERIV : Sous-programme de dérivation pour la détermination de maximum pour les pics d'intensité dans XCALI1.

- YCALI : Programme d'optimisation du paramètre de calibration H pour la fonction trigonométrique qui représente la distorsion dans l'axe Y.

PROGRAMMES DE CALIBRATION

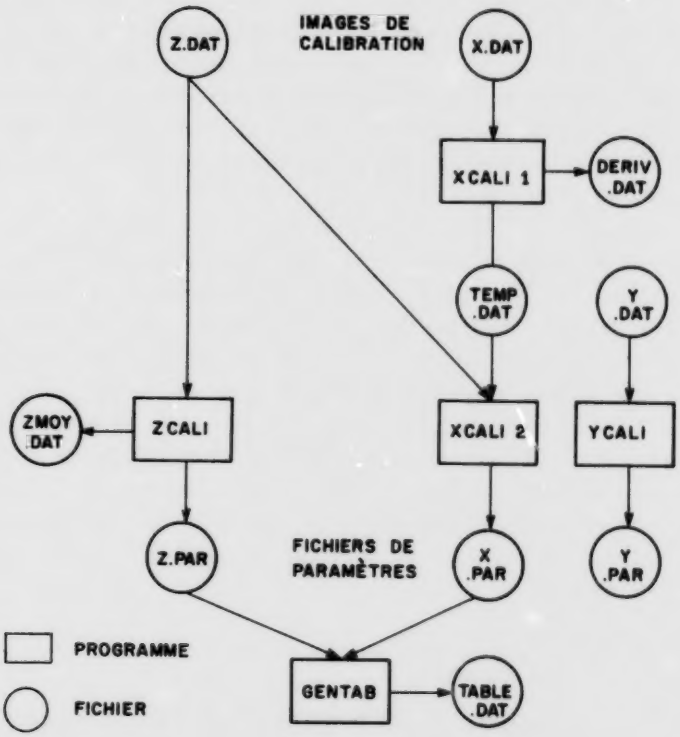


Figure 12. Diagrammes logiques des programmes de calibration.

5. CORRECTION

Que les données soient prisent avec un balayage dans deux axes ou dans un axe avec rotation de l'objet, elles doivent être corrigées d'abord en X et en Z. Cette double correction est faite par le programme XCOR. Si ces données sont prisent avec le balayage en Y, elles peuvent ensuite être corrigées pour les distorsions dans l'axe Y avec YCOR pour donner un fichier à trois tableaux X, Y et Z. Dans le cas de données prisent en rotation, le programme CYLCOR les convertis en coordonnées cartésiennes XYZ dans un fichier de format à trois tableaux comme celui que produit YCOR. Les programmes YCOR et CYLCOR peuvent produire des fichiers de format DPICT avec chacun des tableaux X, Y et Z. Pour pouvoir afficher les données corrigées, il faut les interpoler sur une grille d'espacement fixe qui représente l'écran d'affichage. Les programmes d'interpolation sont XINT et YINT pour les calculs respectifs des axes X et Y. Enfin, si on connait la position des points dans l'image originale que l'on veut mesurer, les programmes CARMES et CYLMES produisent une liste des coordonnées XYZ de chaque point ainsi que les distances qui les séparent.

Tous les programmes de correction produisent des fichiers qui ont le même nom que l'image originale mais dont l'extension est différente et représente la provenance de l'image. La figure 13 illustre sous forme de diagrammes logiques l'ensemble des programmes de correction et de mesures ainsi que leur ordre d'exécution.

5.1 Correction des distorsions en X et en Z

Si on veut faire la correction d'une image complète, la première étape est la même quelle que soit le type d'image. Il s'agit de corriger les données pour l'axe de balayage X et la distance Z. Pour chaque pixel de l'image originale, on connait sa position dans l'image (θ) ainsi que la lecture de la barrette CCD (D). Comme il est expliqué dans la troisième partie, ces deux valeurs sont les paramètres d'entrée de deux fonctions SPLINE qui permettent l'interpolation des coordonnées corrigées X et Z' pour chaque point. Le sous-programme d'interpolation est BISP fournit avec l'ensemble de sous-programmes SURFAC1 (6). Il faut fournir au programme les noms des fichiers de paramètres des fonctions SPLINE.

Le fichier résultant possède l'extension XCO et contient dans chaque registre les coordonnées X et Z de tous les points valides d'une ligne de l'image originale. Le fichier est de format DPICT mais ne peut pas être affiché puisqu'il contient la coordonnnée X. Par contre, les données X et Z peuvent être lue avec le programme LECTURE1. Comme les données sont maintenant corrigées, on connait le facteur d'échelle Z, (SCL) soit la valeur par laquelle il faut diviser les données pour obtenir des mesures

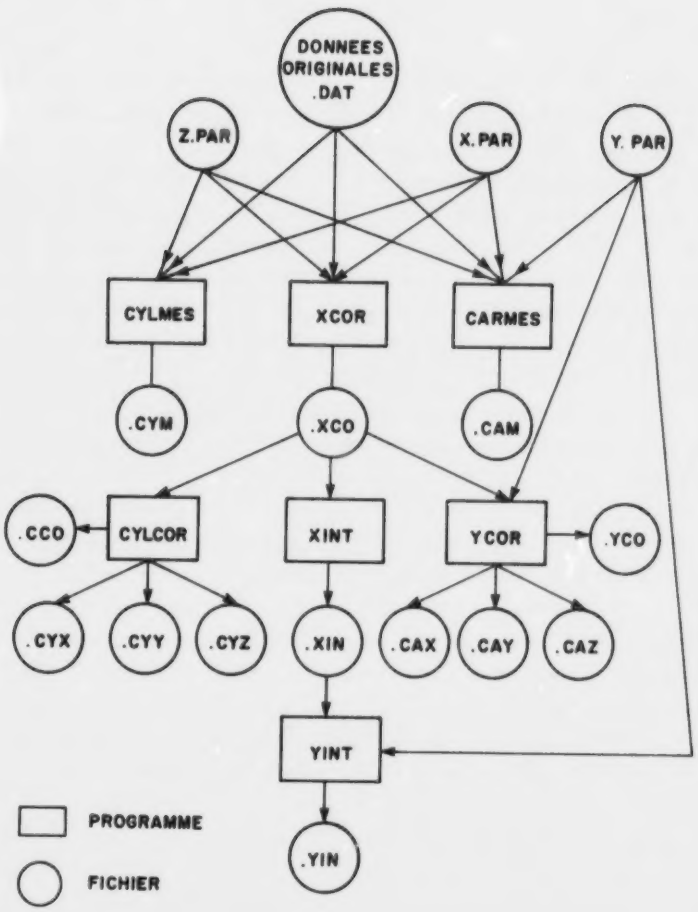


Figure 13. Diagrammes logiques des programmes de correction, de transformation et de mesure.

dans les unités utilisés lors de la calibration. De même, le programme écrit dans le premier registre de l'image corrigée les valeurs minimum et maximum de l'axe X. Le programme LECENT fait une lecture des fichiers de format DPICT et affiche tous les paramètres qui sont écrits dans le premier registre.

5.2 Correction des distorsions en Y

La distorsion causée par le balayage dans l'axe Y peut être corrigée en utilisant les formules vues dans la partie 4.3. Les paramètres optimisés de la fonction trigonométrique sont lues à partir du fichier créé lors de la calibration en Y. Le programme YCOR produit un nouveau fichier d'images qui contient les coordonnées XYZ de chaque point sous le format discuté en section 4.1. Ce format contient dans le premier registre la même information que celui de DPICT et dans les trois registres suivants,

les coordonnées X, Y et Z sous forme de trois tableaux. Les points non valides sont remplacés par des coordonnées zéro dans les trois tableaux. Ce fichier possède l'extension YCO.

Aussi, on peut créer trois images de format DPICT correspondant à chacun des tableaux. Les tableaux sont de mêmes dimensions que l'image originale et on peut retrouver les données corrigées de chaque point de l'image originale aux mêmes coordonnées dans les images X, Y et Z. Les points non valides dans l'image originale sont remplacés par des zéros dans chacun des tableaux X, Y et Z, et les fichiers créés portent les extensions CAX, CAY et CAZ respectivement.

5.3 Transformation des coordonnées cylindriques à cartésiennes

Les fichiers qui sont pris avec rotation de la table tournante peuvent être convertis en fichiers de format à trois tableaux de coordonnées cartésiennes XYZ avec le programme CYLCOR. Ce programme utilise les données qui sont déjà corrigées pour les axes X et Z (extension XCO), et fait la transformation pour trouver la coordonnée Y et transformer la coordonnée Z. Le fichier résultant possède l'extension CCO. Les transformations sont les suivantes :

Si $Z' > R$

[11] $Y = (Z' - R) \sin \alpha$

[12] $Z = (Z' - R) \cos \alpha$

Si $R > Z'$

[13] $Y = (R - ') \sin (\alpha + 180°)$

[14] $Z = (R - ') \cos (\alpha + 180°)$

où

R est la position de l'axe de rotation dans l'axe Z.
Z' est la position corrigée de la surface avant la transformation.
α est l'angle de la table tournante pour la ligne considérée.

La valeur de référence R est la position de l'axe de rotation de la table tournante dans l'axe Z. On peut approximer la valeur de R en faisant une moyenne des lignes de l'image de l'axe de rotation (voir partie 3.3) avec le programme MOYENNE1, et déterminer la meilleure distance en lisant

ce dernier fichier avec LECTURE1. Pour cette transformation, il faut aussi connaître l'angle de la table entre chaque ligne de balayage ($\Delta\alpha$).

Les fichiers produits sont de même format que ceux que produit YCOR. Le fichier à trois tableaux X, Y et Z porte l'extension CCO. Les trois autres finissent par CYX, CYY et CYZ.

5.4. Interpolation sur une grille pour l'affichage graphique

Les données corrigées dans les fichiers précédents (YCO et CCO) sont explicites, c'est-à-dire que les coordonnées X, Y et Z du point apparaissent dans le fichier. Les programmes d'affichages graphiques tels que AFFI1 ou DPICT utilisent des fichiers de coordonnées implicites, soit un tableau de données tri-dimensionnelles seulement (Z) dont la position représente les dimensions X et Y. On présume que les points sont espacés également les uns des autres, ce qui est le cas sur un écran vidéo.

Or les données corrigées ne sont pas également espacées. C'est pourquoi on ne peut pas simplement afficher les données tri-dimensionnelles telles qu'elles se présentent après la correction. Les programmes XINT et YINT font une interpolation linéaire entre les données des fichiers déjà corrigés, et ce dans les axes X et Y respectivement. Le programme XINT se sert des fichiers ayant l'extension XCO, pour créer d'autres fichiers qui finissent par XIN. Si l'image est prise avec deux balayages, le programme YINT reprend ce dernier fichier XIN pour l'interpoler dans l'axe Y et produire un fichier dont l'extension est YIN. La figure 14 est une image brute non corrigée affichée en intensité et la figure 15 est la même image affichée en isométrie. Le même fichier est ensuite corrigé en X, Y et Z par XINT et par YINT pour produire le fichier YIN de la figure 16.

L'algorithme d'interpolation des deux programmes est identique mais exécuté dans leur axe respectif. Par exemple, en connaissant les valeurs minimum et maximum de toute l'image pour l'axe X, il est possible de connaître la dimension de l'incrément X en divisant la différence entre ces deux valeurs extrêmes par le nombre de points désirés dans l'image interpolée. Ensuite, en partant de la valeur minimum, on lui additionne la valeur de l'incrément en X, ce qui nous donne à chaque addition la valeur de X pour laquelle on veut interpoler celle de Z. Il s'agit maintenant, pour chacun de ces X de trouver les deux points dans l'image corrigée, le précédent et le suivant, qui sont le plus près dans l'axe X du point que l'on veut interpoler. Cette interpolation est faite le long de chaque ligne où ϕ est constant. L'interpolation est linéaire entre ces deux valeurs choisies et le résultat est gardé dans un fichier de coordonnées implicite. Dans ce fichier implicite, on y conserve toute l'information sur les facteurs d'échelle et les valeurs minimums et maximums des

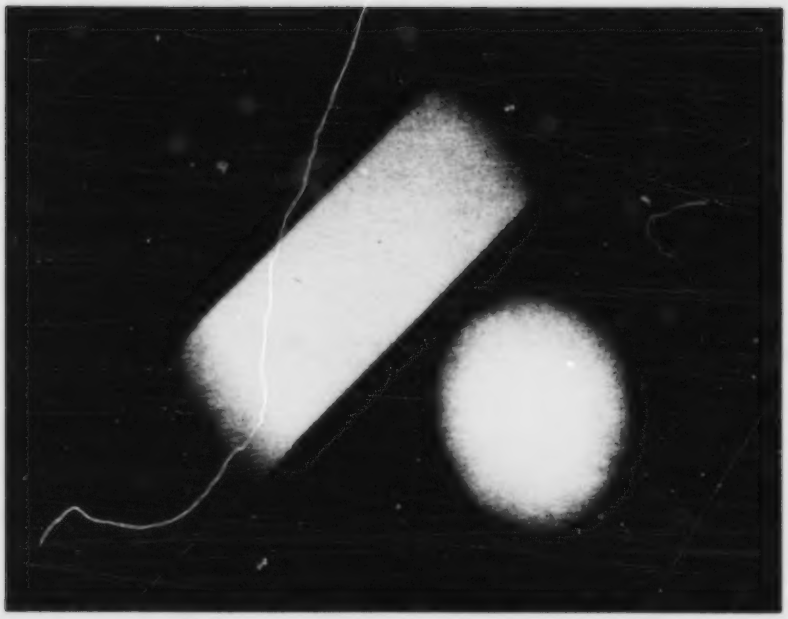


Figure 14. Représentation en niveaux de gris d'une image non corrigée de petits blocs usinés en aluminum sur un fond plat.

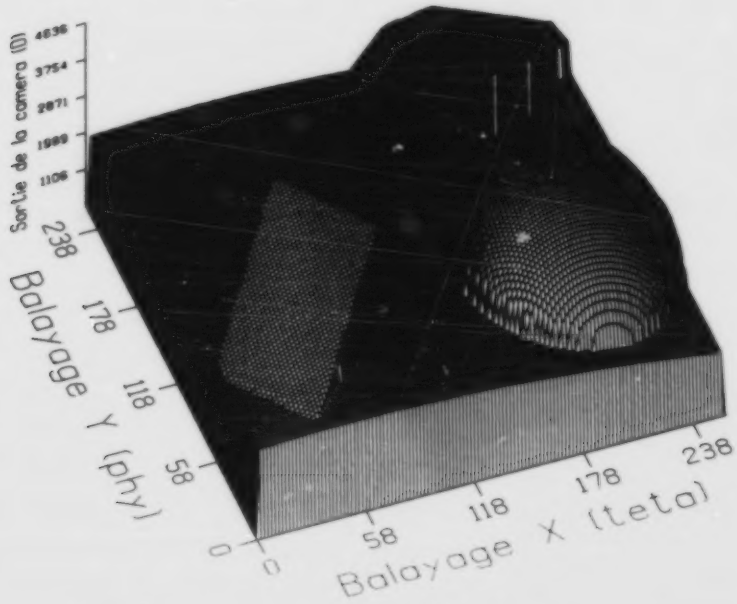


Figure 15. Représentation en isométrie de l'image non corrigée de la figure 14.

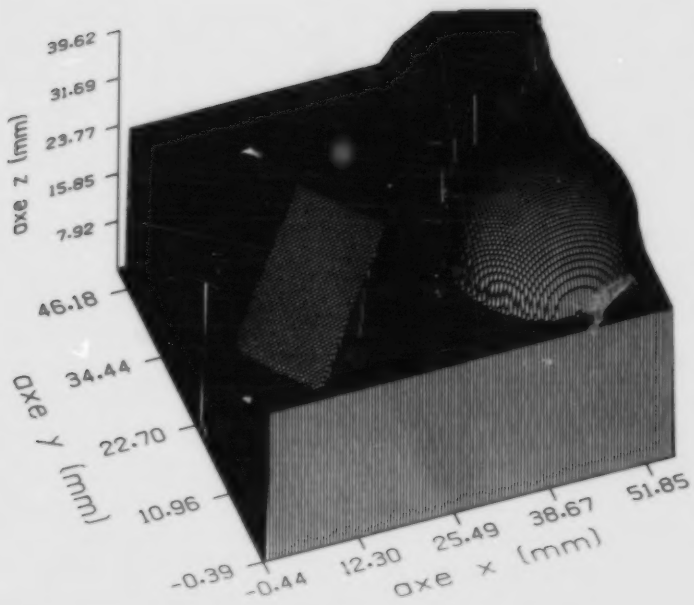


Figure 16. Représentation en isométrie de l'image de la figure 14, corrigée pour les distorsions en X, Y et Z.

balayages X et Y. Ces fichiers peuvent aussi être lues avec les programmes LECTURE1 pour les données, et LECENT pour l'entête seulement.

Pour faire la soustraction de deux images, celles-ci doivent être interpolées entre les mêmes bornes. Il est possible de choisir interactivement les bornes avec les programmes d'interpolation XINT1 et YINT1. Les programmes d'interpolation mettent des valeurs nulles si les valeurs à interpoler sont situées avant les valeurs minimums ou après les valeurs maximums du fichier initial. Il faut donc s'assurer que les bornes de l'interpolation choisis soit à l'intérieur des minimums et maximums de l'image à interpoler. Les images peuvent être soustraitent avec le programme SOUSTR1 (annexe F).

5.5. Mesures

Si on veut effectuer des mesures sur l'image et conserver le maximum de précision de la correction, on ne peut utiliser de fichiers interpolés. La meilleure méthode consiste à ne corriger que les points que l'on veut mesurer et d'afficher les résultats de ces mesures seulement. Deux programmes ont été développés pour faire des mesures sur des fichiers à double balayage et sur d'autres fichiers pris avec rotation de la table tournante. Les points à corriger peuvent être choisis avec le programme d'affichage

AFFI (partie 5.6) sur l'image originale (DAT). On prend en note les coordonnées des points à mesurer.

5.5.1. *Images de coordonnées cartésiennes*

Le programme de mesure pour les images à double balayage est CARMES. La correction pour les coordonnées choisis se fait de la même façon qu'avec la correction des images complètes. Il faut fournir les noms des fichiers de paramètres des fonctions SPLINE ainsi que de la fonction trigonométrique pour la correction en Y. Un tableau est écrit à la fin qui donne la liste des coordonnées des points choisis et des distances tri-dimensionnelles entre chacune d'elles. Un deuxième tableau est créé sous le nom du fichier original avec l'extension CAM. Ce tableau peut être imprimé si on veut garder les résultats à la fin des mesures (tableau 2). Les distances sont données dans les mêmes unités que ceux de la calibration.

Tableau de mesures effectuées sur des points corrigés de l'image originale de la figure 3

```
Measurement between 6 points

Point # 1   x = 135.0698  y = -61.4743  z = 126.0157  px = 100  py = 110
Point # 2   x = 267.9426  y = -29.2688  z = 127.9175  px = 239  py = 150
Point # 3   x =  55.1934  y = -71.0809  z = 111.5670  px =  20  py = 100
Point # 4   *** INVALID DATA FOR THIS POINT ***       px =  20  py =   2
Point # 5   x = 230.7536  y = -80.3494  z = 143.0749  px = 200  py =  84
Point # 6   x =  87.0786  y = -46.8925  z = 127.4129  px =  50  py = 128

The distance between the points 2 and 1 is 136.7332 unit(s)

The distance between the points 3 and 1 is  81.7392 unit(s)
The distance between the points 3 and 2 is 217.4346 unit(s)

The distance between the points 5 and 1 is  99.0084 unit(s)
The distance between the points 5 and 2 is  64.9769 unit(s)
The distance between the points 5 and 3 is 178.6058 unit(s)

The distance between the points 6 and 1 is  50.1771 unit(s)
The distance between the points 6 and 2 is 181.7213 unit(s)
The distance between the points 6 and 3 is  43.0446 unit(s)
The distance between the points 6 and 5 is 148.3482 unit(s)
```

5.5.2. *Images de coordonnées cylindriques*

De la même façon, le programme CYLMES effectue la correction des points que l'on aura identifiés sur l'image originale. Dans ce cas, on

doit entrer la valeur de l'angle α de rotation de la table tournante entre deux lignes de balayage en X, ainsi que la position R en unités de l'axe de rotation de la table (partie 5.3). Le programme produit aussi un fichier de la liste des mesures entre les points dont le nom combine celui de l'image originale et l'extension CYM.

5.6 Affichage et mesures

Les programmes d'affichage décrit dans cette section sont ceux développés pour le système graphique TRAPIX de RECOGNITION CONCEPTS INC. et qui utilisent les sous-programmes de base de Theory and Applications Unlimited Corporation (TAU). Ces programmes sont structurés sous forme de menus principaux qui appellent des sous-programmes spécifiques qui à leur tour appellent des fonctions de base de la librairie de TAU.

L'affichage se fait sur un tableau de 512 × 512 éléments graphiques ou pixels. Le système peut contenir simultanément six tableaux différents. Chacun de ces tableaux est un sous-canal d'un canal principal. Il y a trois canaux principaux et deux sous-canaux par canal. On les désigne par un nombre de deux chiffres dont le premier représente le canal et le deuxième le sous-canal. Les tableaux disponibles sont par conséquent désignés par les nombres 00, 01, 10, 11, 20 et 21.

Il y a deux types d'affichage qui ont été développés, soit en tons de gris et en isométrie. Le premier code l'information tri-dimensionnelle en niveaux de gris ou le blanc représente la valeur le plus élevée soit la plus proche du capteur et le noir la plus éloignée. Le deuxième mode d'affichage est la représentation en isométrie où l'information est codée en hauteur où chaque ligne de balayage est déphasée par rapport à la précédente (figs. 21, 22 et 23). Aussi, les lignes dont l'information est inférieure à celle qui la précède ne sont pas tracés pour ces régions, ce qui en fait une représentation avec lignes cachées. Le programme AFFI1 combine les deux modes dans un seul programme général qui utilise les données de format DPICT. Le programme AFFI utilise les données avant la correction (DAT). Il est utilisé pour vérifier la validité des données tout de suite après l'enregistrement (bit de saturation et de validation).

Dans les deux modes d'affichage, le nombre de niveaux de gris disponible est de 256. L'information codée en réels est tronquée de sa partie fractionnaire pour être mise dans des entiers de 16 bits. En spécifiant le BIT DE DÉPART(6), on détermine le bit à partir duquel les 8 bits de l'affichage seront pris dans le mot de 16 bits. Par exemple, en choisissant le bit 2, les 8 bits affichés seront 2, 3, 4, 5, 6, 7, 8 et 9 (fig. 17). Aussi, en spécifiant le BIT DE DÉPART à -1, le gain de l'affichage est automatique et on a une variation de niveaux de gris qui est optimum pour les

Figure 17. Affichage en niveaux de gris à gain fixe avec le bit de départ à 2, de l'image de la figure 3.

données affichées (fig. 3). Dans ce cas, la tronquation en un entier est faite après le calcul de gain automatique.

L'information sur les dimensions de l'images est lue du premier registre de l'image et est constamment affichée sur le menu. L'affichage peut se faire sur l'image complète ou sur une partie seulement. La région choisie est alors spécifiée par les POSITION INITIALE et FINALE de l'affichage (options 3 et 4). On peut aussi choisir la POSITION DE DÉPART (5) de l'affichage sur l'écran. Le choix du canal se fait par l'OPTIONS SPÉCIALES (7) qui affiche un deuxième menu. On peut décider alors de CHANGER (D) de canal, ou de l'EFFACER (E), ou d'exécuter un ZOOM (Z) interactif avec lequel on peut changer la position de l'image sur l'écran en temps réel. La position du pixel de départ de l'affichage sur l'écran peut aussi être choisie interactivement par l'option ORIGINE (O) qui l'affiche par un curseur rouge.

L'option du CURSEUR (X) permet de cadrer une partie de l'image qui est sur l'écran et refaire l'affichage pour cette portion seulement. On sort de cette option en tapant E pour EXIT, où on peut connaître les coordonnées graphiques avec S. Le gain automatique affiche le maximum de lontraste entre les deux curseurs graphiques. Cette option est très pratique pour faire ressortir les détails d'une partie de l'image (fig. 18). Les curseurs sont utilisés aussi pour la mesure entre deux points de

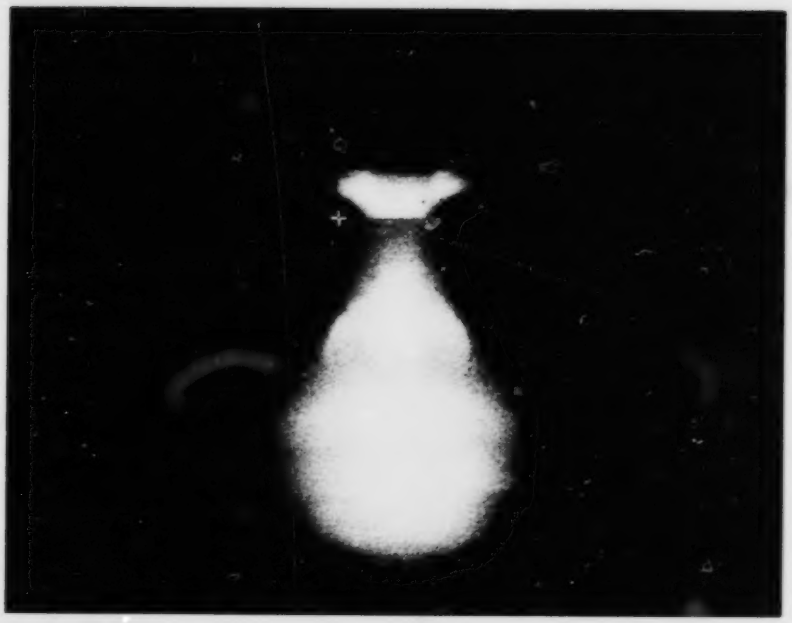


Figure 18. Utilisation de curseurs pour cadrer une partie de l'image sur laquelle est appliquée le gain automatique.

l'image. Après avoir positionné les deux curseurs sur les pixels à mesurer, l'option de MESURE (M) affiche leurs coordonnées dans les trois axes, convertis en unités avec les facteurs d'échelle. La différence entre ces coordonnées est aussi affichée sur l'écran graphique. Ces différences sont données entre les curseurs pour chacun des trois axes X, Y et Z, pour la distance dans un plan bi-dimensionnel 2D et dans le volume tri-dimensionnel (fig. 19).

L'affichage en isométrie peut se faire avec des lignes tracées horizontalement ou verticalement (figs. 20 et 21). Les deux tracés peuvent être utilisés pour produire un grillage. Si on veut représenter une image contenant beaucoup de points par un grillage sans que celui-ci paraisse trop dense, on peut prendre seulement quelques points et quelques lignes espacées uniformément avec les options correspondantes (fig. 22). Aussi, on peut avec les curseurs précédents, choisir une portion de l'image que l'on peut amplifier dans les trois axes pour une meilleure représentation graphique (fig. 23). Le facteur en Z peut être utilisé pour atténuer puisqu'il peut être fractionnaire. L'option DÉCALAGE (9) permet de simuler un genre de rotation qui est en fait un décalage horizontal entre chaque ligne du tracé. Ce décalage peut aussi bien être positif que négatif.

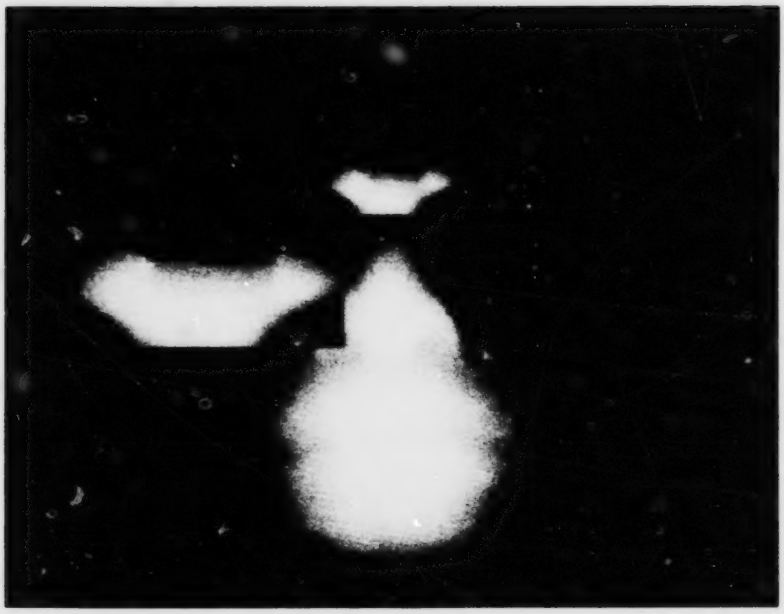


Figure 19. Affichage des mesures tri-dimensionnelles effectuées sur les données entre les deux curseurs graphiques, et agrandissement de la zone encadrée.

Figure 20. Affichage en isométrie avec lignes horizontales.

Figure 21. Affichage en isométrie avec lignes verticales.

Figure 22. Affichage en isométrie avec les lignes dans les deux axes et sous-échantillonage pour diminuer la densité. Aussi, utilisation du décalage négatif.

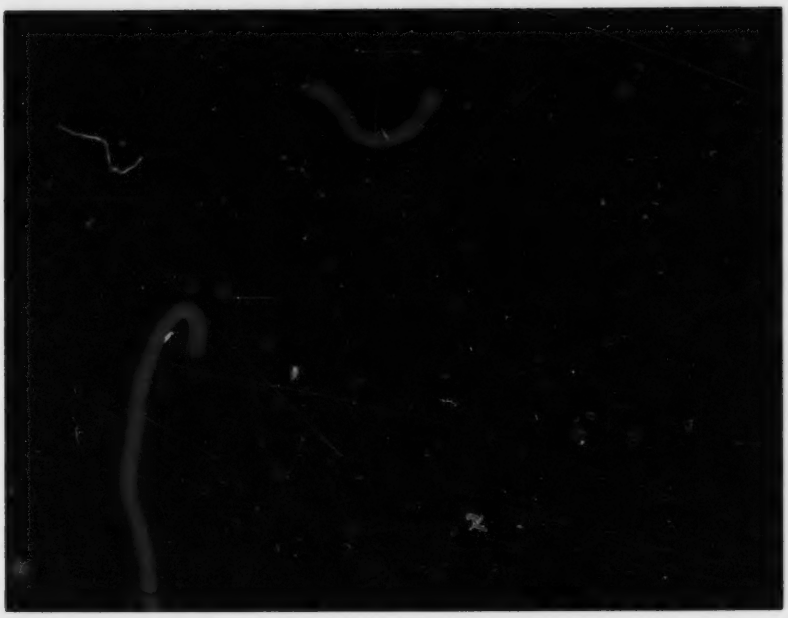


Figure 23. Affichage en isométrie de la portion de l'image cadrée de la figure 19 et une expansion dans les trois axes.

Les programmes d'affichage graphique de DPICT même peuvent être utilisés sur les données corrigées. Par exemple, le programme DPICT MESH a produit la vue en isométrie des figures 15 et 16 dont la résolution graphique est supérieure, ce programme fonctionnant avec une imprimante à laser à haute résolution. Le temps requis pour cet affichage est de beaucoup supérieur à celui pris par le programme AFFI1 (15 min au lieu de 10 s). Mais la vue est plus réelle puisqu'il est possible d'effectuer de vraies rotations et non pas de simples décalages entre les lignes (fig. 16), ainsi que des perspectives avec point de fuite.

Enfin, il existe une série d'autres programmes pour modifier les données d'un fichier original ou de format DPICT. Les programmes utilisant le format DPICT se terminent par 1 (annexe F).

Programmes de correction (annexe C) :

- XCOR : Programme de correction des données brutes à partir des fonctions spline FZ et FX2.
- YCOR : Programme de correction des données dans l'axe Y pour les images avec double balayage avec création d'un fichier à trois tableaux X, Y et Z.

- YCOR1 : Programme similaire à YCOR mais avec création d'un fichier à quatre tableaux binaire PPICT ou ASCI1
- XINT : Programme d'interpolation linéaire pour l'affichage graphique des données corrigées par XCOR.
- XINT1 : Programme similaire à XINT avec détermination manuelle des bornes.
- YINT : Programme d'interpolation linéaire pour l'affichage graphique des données prisent avec balayage en Y.
- YINT1 : Programme similaire à YINT avec détermination manuelle des bornes.
- CYLCOR : Programme de transformation en coordonnées cartésiennes, des données corrigée en X, Y et Z, prisent avec rotation de la table tournante, produisant un fichier à trois tableaux X, Y et Z.
- CYLCOR1 : Programme similaire a CYLCOR mais avec creation d'un fichier binaire DPICT ou ASCI1.

Programmes de mesures (annexe D) :

- CARMES : Programme de mesures sur des coordonnées de l'image originale avec balayage dans deux axes.
- CYLMES : Programme de mesures sur des coordonnées de l'image originale avec rotation de la table tournante.

Programmes d'affichage (annexe E) :

- AFFI : Programme général d'affichage graphique en intensité et en isométrie, sur écran pour les fichiers originaux
- AFFI1 : Programme général d'affichage graphique en intensité et en isométrie, et de mesures pour les fichiers de format DPICT
- INTEN : Sous-programme d'affichage en niveaux de gris pour le programme AFFI
- INTEN1 : Sous-programme d'affichage en niveaux de gris utilisé par le programme AFFI2
- ISOINT : Sous-programme d'affichage en isométrie pour le programme AFFI
- ISOREAL : Sous-programme d'affichage en isométrie pour le programme AFFI1
- LECCOOR : Sous-programme de lecture des coordonnées et de mesures avec les facteurs d'échelle de la calibration
- AFFICOOR : Sous-programme d'affichage des mesures faites avec LECCOOR sur l'écran graphique

Programmes de transformation et de lecture (annexe F) :

- REDUIT : Programme de réduction d'une image dans les axes X et Y par un facteur entier.
- EXPAND : Programme d'expansion d'une image dans les axes X et Y par un facteur entier.
- MULTIPLIE: Programme qui multiplie les données
- OFFSET : Programme qui additionne aux données une valeur réelle.
- JONCTION : Programme pour joindre plusieurs images ensemble
- SEPAR : Programme pour séparer les images tri-dimensionelles et de réflectivité prises en alternance
- TRANSVAR : Programme pour convertir un fichier dont la longueur des registres est fixée à 1024 octets, à des registres de longueur variable
- MOYLIGNE : Programme pour effectuer la moyenne des lignes d'une image entre elles.
- SOUSTR : Programme de soustraction de deux images de même dimension.
- PIVOT : Programme effectuant le pivot d'une image par rapport à son centre sur un des trois axes X, Y et Z.
- LECTURE : Programme de lecture des données d'une image originale.
- LECTURE1 : Programme de lecture des données d'une image de format DPICT.
- LECTURE2 : Programme de lecture pour les fichiers de format binaire DPICT XYZ.
- LECTURE3 : Programme de lecture des fichiers de format binaire DPICT XYZ, mais dans l'axe perpendiculaire à celui de LECTURE2.
- LECENT : Programme de lecture du premier registre des fichiers de format DPICT, pour connaître les dimensions et les facteurs d'échelle d'une image

REMERCIEMENT

L'auteur tient à remercier Marc Rioux pour l'aide apporté lors de la rédaction de la première partie traitant du principe optique ainsi que Français Blais et Pierre Boulanger pour les discussions concernant les algorithmes de calibration et leur programmation.

RÉFÉRENCES

1. M. Rioux. Laser range finder based on synchronized scanners. Appl. Opt. 23(21): 3837-3844; 1984.

2. M. Rioux. Compact, high speed 3-D vision sensor for robotic applications. Proc. 2nd Canadian CAD/CAM and Robotics Conference, Toronto (Ontario). 1983. pp. 4.11-4.16.

3. P. Boulanger, M. Rioux, K.B. Evans et L. Rhuhlman. Interface Between a 3-D Laser Scanner and a CAD/CAM System, Fifth Canadian CAD/CAM and Robotics Conference, Toronto (Ontario). 1986. pp. 731-737.

4. F. Bumbaca, F. Blais et M. Rioux. Real-time correction of three-dimensional nonlinearities for a laser rangefinder. Opt. Eng. 25(4): 561-565; 1986.

5. F. Blais et M. Rioux. Real-time numerical peak detector. Signal Process. 11: 145-155; 1986. NRC 26561.

6. P. Dierckx. An Algorithm for surface fitting with spline functions. J. Num. Anal. 1(3): 267-283; 1981.

ANNEXE A

Sources des programmes d'acquisition sur DPS-1

```
CAMPOC1:

        PROCEDURE OPTIONS (MAIN,STACK(1000));
/*

CNRC, Octobre 1983      Luc Cournoyer                   Revisee : 4/9/86

                        Programme CAMPOC
                        ----------------

        Programme pour l'utilisation de la camera tridimensionelle
fabriquee par POCATEC LTEE. Le programme envoi les parametres
necessaires a l'initisalisation de la camera pour les modes de
fonctionnement d'image, de ligne et de point et de faire l'enregistrement
des donnees selon les parametres specifies.

(Version Interactive et sans affichage graphique)

*/
        %INCLUDE 'MACRODCL.LIB';
        %INCLUDE 'STRUC.LIB';
        %REPLACE  PORT BY 'E2'B4;  /* port pour pagination */

        %REPLACE PAGE_PARA BY '0003'B4;  /* NUMERO DE LA PAGE DES PARAMETRES */
        %REPLACE POSITION_PARA BY 'E000'B4;    /* ADRESSE DES PARAMETRES
                                                  DANS LA PAGE */
        %REPLACE PAGE_ACQUIS BY '0003'B4;  /* NUMERO DE LA PAGE */
        %REPLACE POSITION_ACQUIS BY 'E500'B4; /* ADRESSE DU PARAMETRE
                                                  D'ACQUISITION DANS LA PAGE */
        %REPLACE PAGE_DONNEES BY '0002'B4; /* NUMERO DE LA PAGE DES DONNEES */
        %REPLACE POSITION_DONNEES BY '0000'B4;    /* ADRESSE DES DONNEES */

/* POINTER to an array descriptor */

DECLARE
    Array_descriptor            POINTER,
    1    Array_descriptor_structure     BASED (Array_descriptor) ,
         2    A_Dim(2)           BINARY,
         2    A_Logique(2)               POINTER,
         2    A_Addr_ptr         DECIMAL,
         2    A_Precision                BINARY;


DCL  /* Fonctions de systeme */
        poutput entry(bit(8),binary);

DCL
        Free_struct  Entry(pointer),
        ADCONV_P_A  ENTRY (POINTER,POINTER),
        MAPPER ENTRY(POINTER,POINTER,POINTER);

DCL
        soustraction    character(1),
        change          character(1),
```

```
        reponse         character(1),
        choix_angle     character(1),
        COOR (2) BINARY BASED (PT_COOR),
        PT_COOR POINTER;

DCL
        value_binary     binary,
        value_bit_huit   bit(8)    based(pt_value),
        value_as_pointer POINTER   based(pt_value),
        value_in_bit     bit(16)   based (pt_value),
        IMODE            BINARY,
        pt_value         pointer;

        IMODE=0;         /* DEFINIE LE MODE MANUEL */

        pt_value=addr(value_binary);

DCL
        (I,J)            BINARY,
        VALUE BIT(8) BASED;              /* logique(1) ->  */

/* initialization des parametres  */
/* on positionne les structures */

        allocate window_structure set(window);
        allocate map_parms_structure set(map_parms);
        allocate macro_descriptor set(ptmacro);

/* DECLARE LES POINTEURS POUR LE DMA */

DCL
        reset_command           pointer,
        DMA_PARMS               POINTER,
        DMA_MAP                 POINTER,
        PARAMETRES              POINTER,
        DONNEES                 POINTER,
        MOT_RESET               POINTER,
        MOT_ACQUIS              POINTER;

/* DECLARE LES PARAMETRES POUR LA CAMERA */

declare /* fonctions */
        remap entry (BINARY,POINTER),
        setdma entry (POINTER),
        dma entry (FIXED BIN(15),POINTER),
        rampe entry (POINTER,POINTER,BINARY),
        escal0 entry (POINTER,POINTER,BINARY) ,
        ligne0 entry (POINTER,POINTER,BINARY) ,
        ligne1 entry (POINTER,POINTER,BINARY) ,
        POINT ENTRY (POINTER,POINTER,BINARY),
        SETXY entry (POINTER,POINTER) ,
        para1 entry ;

DECLARE  /* STRUCTURE DEFINISSANT LES PARAMETRES DU FICHIER
            A ENVOYER POUR UNE ACQUISITION.  */
```

```
        para_acquis_ptr          pointer,
        para_acquis bit(16) based(para_acquis_ptr);

DCL     clr                      character(8);
        clr = '^[[2J ^[[H';      /* EFFACE L'ECRAN */

DECLARE  /* VARIABLE SERVANT A IDENTIFIER LE DERNIER MODE
            AYANT SERVI A L'IDENTIFICATION DE LA CAMERA. */
        ETAT                     BINARY;

        CALL SETDMA(dma_parms);

        PT_COOR=ADDR(LOGIQUE(1));

/* FAIT LES ALLOCATIONS EN VUE DE TRANSFERTS DMA */

        ALLOCATE WINDOW_STRUCTURE SET(PARAMETRES);
        ALLOCATE WINDOW_STRUCTURE SET(DONNEES);
        ALLOCATE WINDOW_STRUCTURE SET(MOT_ACQUIS);
        ALLOCATE WINDOW_STRUCTURE SET(MOT_RESET);

        UNSPEC(PARAMETRES->LOGIQUE(1))= PAGE_PARA;
        UNSPEC(PARAMETRES->LOGIQUE(2))= POSITION_PARA;

/* Addresses du mot d'acquisition */

        UNSPEC(MOT_ACQUIS->LOGIQUE(1))= PAGE_ACQUIS;
        UNSPEC(MOT_ACQUIS->LOGIQUE(2))= POSITION_ACQUIS;
        unspec(para_acquis_ptr)='E500'b4;
        mot_acquis->dim(1) = 1;
        mot_acquis->dim(2) = 1;

/* ADDRESSES DES DONNEES */

        UNSPEC(DONNEES->LOGIQUE(1))= PAGE_DONNEES;
        UNSPEC(DONNEES->LOGIQUE(2))= POSITION_DONNEES;

/* Affiche les options possibles */
DO WHILE ('1'b);
MENU:
        put skip    list(clr,'                                        ');
        put skip(2) list('                                  CAMERA POCATEC ');

        PUT SKIP(2) LIST('PREPARATION DE LA CAMERA');

        PUT SKIP(2) LIST('      A. Mode image (rampe)                ');
        PUT         LIST('              B. Mode image (escalier)');
        PUT SKIP    LIST('      C. Mode ligne                        ');
        PUT         LIST('                          D. Mode point');
        PUT SKIP    LIST('      E. Nombre de transferts');
        PUT         LIST('                           F. Balayage seulement');

        PUT SKIP(2) LIST('ACQUISITION DES DONNEES');
```

```
        PUT SKIP(2) LIST('      G. Enregistrement (2 bytes/pixel)');
        PUT SKIP    LIST('      H. Enregistrement (4 bytes/pixel)');
        PUT SKIP    LIST('      I. Transfert paralelle 16 bits DMA ');

        put skip    list('      S. Sortir');

        put skip(2) list('Reponse  :   ');
        get list (reponse);

/* Envoi a la sousroutine correspondante. */

        GOTO CHOIX(INDEX('ABCDEFGHIS',reponse));

CHOIX(0):
        do;
        put skip(3) list ('----->> MAUVAISE REPONSE <<-----');
        end;
        goto menu;

CHOIX(1):
        do;
        call remap(1,PARAMETRES);
        PUT SKIP LIST(clr,'MODE INTERACTIF (1) OU MANUEL (0) ? ');
        GET LIST (IMODE);
        if imode=1 then CALL SETXY(PARAMETRES,DONNEES);
        call rampe(PARAMETRES,DONNEES,IMODE);
        etat=1 ;
        end;
        goto menu;

CHOIX(2):
        do;
        call remap(1,PARAMETRES);
        PUT SKIP LIST(clr,'MODE INTERACTIF (1) OU MANUEL (0) ? ');
        GET LIST (IMODE);
        if imode=1 then CALL SETXY(PARAMETRES,DONNEES);
        call escal0(PARAMETRES,DONNEES,IMODE);
        etat=2 ;
        end;
        goto menu;

CHOIX(3):
        do;
        call remap(1,PARAMETRES);
        PUT SKIP LIST(clr,'MODE INTERACTIF (1) OU MANUEL (0) ? ');
        GET LIST (IMODE);
        IF IMODE=1 THEN CALL SETXY(PARAMETRES,DONNEES);
        put skip list('Angle aigus ou obtus (<45=0,>45=1)');
        get skip list(choix_angle);
        if (choix_angle=0) then call ligne0(PARAMETRES,DONNEES,IMODE);
        else call ligne1(PARAMETRES,DONNEES,IMODE);
        etat=3 ;
        end;
        goto menu;
```

```
CHOIX(4):
        do;
        call remap(1,PARAMETRES);
        PUT SKIP LIST(clr,'MODE INTERACTIF (1) OU MANUEL (0) ? ');
        GET LIST (IMODE);
        IF IMODE=1 THEN call SETXY(PARAMETRES,DONNEES);
        CALL POINT(PARAMETRES,DONNEES,IMODE);
        etat=4 ;
        end;
        goto menu;

CHOIX(5):
        do;
        put skip(2) list(clr,'Nombre de transferts desires :');
        get list(DONNEES->DIM(2));
        end;
        goto menu;

CHOIX(6):
        do;
        para_acquis = '0100'b4;
        call remap (1,MOT_ACQUIS);
        call dma (3,MOT_ACQUIS);
        end;
        goto menu;

CHOIX(7):
        do;
        para_acquis = '0100'b4;
        call remap (1,MOT_ACQUIS);
        call dma (3,MOT_ACQUIS);
        call remap (1,DONNEES);
        call dma (2,DONNEES);
        end;
        goto menu;

CHOIX(8):
        do;
        para_acquis = '0100'b4;
        call remap (1,MOT_ACQUIS);
        call dma (3,MOT_ACQUIS);
        DONNEES->DIM(2)=DONNEES->DIM(2) * 2;
        call remap (1,DONNEES);
        call dma (2,DONNEES);
        DONNEES->DIM(2)=DONNEES->DIM(2) / 2;
        end;
        goto menu;

CHOIX(9):
        do;
        put  list (clr);
        call para1();
        end;
        goto menu;
```

```
CHOIX(10):
        do;
        STOP;
        end;

end;    /* de la boucle while */

END CAMPOC1;
```

```
DIGIS1:

        PROCEDURE OPTIONS (MAIN,STACK(1000));
/*

CNRC, Octobre 1984      Pierre Boulanger                Revisee : 4/9/86


                        Programme DIGIS0
                        ----------------
        Programme de digitalisation d'un solide utilisant la camera
tridimensionelle fabriquee par POCATEC LTEE et une table tournante
programmable. La methode consiste a faire tourner l'object face a
un digitaliseur de profile et ainsi obtenir les coordonnees x,z,alpha
de toute sa surface visible.

(Version Interactive et sans affichage graphique)

*/
        %INCLUDE 'MACRODCL.LIB';
        %INCLUDE 'STRUC.LIB';
        %REPLACE  PORT BY 'E2'B4;  /* port pour pagination */

        %REPLACE PAGE_PARA BY '0003'B4;  /* NUMERO DE LA PAGE DES PARAMETRES */
        %REPLACE POSITION_PARA BY 'E000'B4;    /* ADRESSE DES PARAMETRES
                                                  DANS LA PAGE */
        %REPLACE PAGE_ACQUIS BY '0003'B4;  /* NUMERO DE LA PAGE */
        %REPLACE POSITION_ACQUIS BY 'E500'B4; /* ADRESSE DU PARAMETRE
                                                  D'ACQUISITION DANS LA PAGE */
        %REPLACE PAGE_DONNEES BY '0002'B4; /* NUMERO DE LA PAGE DES DONNEES */
        %REPLACE POSITION_DONNEES BY '0000'B4;    /* ADRESSE DES DONNEES */

/* POINTER to an array descriptor */

DECLARE
    Array_descriptor            POINTER,
    1    Array_descriptor_structure     BASED (Array_descriptor) ,
         2    A_Dim(2)           BINARY,
         2    A_Logique(2)               POINTER,
         2    A_Addr_ptr         DECIMAL,
         2    A_Precision                BINARY;

DCL
        Free_struct  Entry(pointer),
        ADCONV_P_A  ENTRY (POINTER,POINTER),
        MAPPER ENTRY(POINTER,POINTER,POINTER);

DCL
        soustraction    character(1),
        change          character(1),
        reponse         character(1),
        COOR (2) BINARY BASED (PT_COOR),
        PT_COOR POINTER;
```

```
DCL
        value_binary      binary,
        value_bit_huit    bit(8)     based(pt_value),
        value_as_pointer  POINTER    based(pt_value),
        value_in_bit      bit(16)    based (pt_value),
        IMODE             BINARY,
        pt_value          pointer;

        IMODE=0;          /* DEFINIE LE MODE MANUEL */

        pt_value=addr(value_binary);

DCL
        (I,J)             BINARY,
        VALUE BIT(8) BASED;              /* logique(1) ->  */

/* initialization des parametres  */
/* on positionne les structures */

        allocate display_list set(disp_ptr);
        allocate window_structure set(window);
        allocate map_parms_structure set(map_parms);
        allocate macro_descriptor set(ptmacro);

/* DECLARE LES POINTEURS POUR LE DMA */

DCL
        reset_command             pointer,
        DMA_PARMS                 POINTER,
        DMA_MAP                   POINTER,
        PARAMETRES                POINTER,
        DONNEES                   POINTER,
        MOT_RESET                 POINTER,
        MOT_ACQUIS                POINTER;


/* DECLARE LES PARAMETRES POUR LA CAMERA */

declare /* fonctions */
        remap entry (BINARY,POINTER),
        setdma entry (POINTER),
        dma entry (FIXED BIN(15),POINTER),
        defacq entry (POINTER,POINTER,BINARY) ,
        SETXY entry (POINTER,POINTER) ,
        acqui1 entry (pointer,pointer),
        para1 entry ;

DECLARE  /* STRUCTURE DEFINISSANT LES PARAMETRES DU FICHIER
            A ENVOYER POUR UNE ACQUISITION.  */
        para_acquis_ptr          pointer,
        para_acquis bit(16) based(para_acquis_ptr);

DECLARE  /* STRUCTURE DEFINISSANT LES PARAMETRES
            DU FICHIER A ENVOYER POUR UN RESET  */
```

```
        para_reset_ptr          pointer,
        para_reset bit(16) based(para_reset_ptr),
        clr                     character(8);

        clr='^[[2J ^[[H';/*EFFACE L'ECRAN d'un VT102*/

        CALL SETDMA(dma_parms);

        PT_COOR=ADDR(LOGIQUE(1));

/* FAIT LES ALLOCATIONS EN VUE DE TRANSFERTS DMA */

        ALLOCATE WINDOW_STRUCTURE SET(PARAMETRES);
        ALLOCATE WINDOW_STRUCTURE SET(DONNEES);
        ALLOCATE WINDOW_STRUCTURE SET(MOT_ACQUIS);
        ALLOCATE WINDOW_STRUCTURE SET(MOT_RESET);

        UNSPEC(PARAMETRES->LOGIQUE(1))= PAGE_PARA;
        UNSPEC(PARAMETRES->LOGIQUE(2))= POSITION_PARA;

/* Addresses du mot d'acquisition */

        UNSPEC(MOT_ACQUIS->LOGIQUE(1))= PAGE_ACQUIS;
        UNSPEC(MOT_ACQUIS->LOGIQUE(2))= POSITION_ACQUIS;
        unspec(para_acquis_ptr)='E500'b4;
        mot_acquis->dim(1) = 1;
        mot_acquis->dim(2) = 1;

/* ADDRESSES DES DONNEES */

        UNSPEC(DONNEES->LOGIQUE(1))= PAGE_DONNEES;
        UNSPEC(DONNEES->LOGIQUE(2))= POSITION_DONNEES;

/* Affiche les options possibles */
DO WHILE ('1'b);
MENU:
        put list(clr);
        put SKIP        list ('                    PROGRAMME DIGIS0 VERSION 1.1');

        PUT SKIP(2)     LIST('PREPARATION DE LA CAMERA');
        PUT SKIP        LIST('    A. DEFINITION DE LA LIGNE DE BALAYAGE');
        PUT             LIST(' (MANUEL) ');
        PUT SKIP        LIST('    B. DEFINITION DE LA LIGNE DE BALAYAGE');
        PUT             LIST(' (INTERACTIF)');
        PUT SKIP        LIST('    C. DEBUTE L''ACQUISITION');
        PUT SKIP        LIST('    I. TRANSFERT PARALELLE 16 bits DMA ');
        PUT SKIP        LIST('    S. SORTIR');
        PUT SKIP(2)     LIST(' REPONSE   :   ');
        GET             LIST(REPONSE);

/* Envoi a la sousroutine correspondante. */

        GOTO CHOIX(INDEX('ABCIS',reponse));
choix(1):
```

```
Rampe:
        PROCEDURE (PARAMETRES,DONNEES,IMODE) ;

        %INCLUDE 'STRUC.LIB';

DCL     IMODE               BINARY;

DCL     dma entry (fixed bin(15),pointer);

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES
            A ENVOYER POUR UNE IMAGE AVEC RAMPE. */
        para_pointer            pointer,
    1   para_source based(para_pointer),
        2 teta_initiale         binary,
        2 phy_initiale          binary,
        2 teta_final            binary,
        2 phy_final             BINARY,
        2 vitesse_teta_aller    binary,
        2 vitesse_teta_retour   binary,
        2 increment_teta        binary,
        2 vitesse_phy_aller     binary,
        2 nombre_de_ligne       binary;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES MODIFIEES A ENVOYER */
        para_rampe_ptr          pointer,
    1   para_rampe based(para_rampe_ptr),
        2 controle0             bit(16),
        2 para1                 binary,
        2 para2                 binary,
        2 para3                 binary,
        2 para4                 binary,
        2 para5                 binary,
        2 para6                 binary,
        2 para7                 binary,
        2 para8                 binary,
        2 para9                 binary,
        2 reset0                bit(16),
        2 reset1                bit(16),
        2 controle1             bit(16);

DCL     rampepar                file;

DCL     (i,NC,NL)               BINARY,
        (PARAMETRES,DONNEES)    POINTER;

DCL
        curr_wrd        fixed bin(15),
        pcurr_wrd       fixed decimal(6),
        pt_valeur       pointer,
        valeur_bit      bit(16) based (pt_valeur) ,
        valeur_binary   BINARY  based (pt_valeur),
        valeur_as_pointer       POINTER;

DCL     CLR             CHARACTER(8),
        NUMERO          CHARACTER(1);

        CLR = '^[[2J ^[[H' ;    /* EFFACE L'ECRAN */
```

```
        do;
        CALL REMAP(1,PARAMETRES);
        CALL DEFACQ(PARAMETRES,DONNEES,0);
        END;
        GOTO MENU;
CHOIX(2):
        DO;
        CALL REMAP(1,PARAMETRES);
        CALL SETXY(PARAMETRES,DONNEES);
        CALL DEFACQ(PARAMETRES,DONNEES,1);
        END;
        GOTO MENU;
CHOIX(3):
        DO;
        CALL REMAP(1,PARAMETRES);
        CALL ACQUI1(DONNEES,MOT_ACQUIS);
        END;
        GOTO MENU;
CHOIX(4):
        do;
        put  list (clr);
        call para1();
        end;
        goto menu;
CHOIX(5):
        do;
        STOP;
        end;
CHOIX(0):
        do;
        put skip(3) list (clr,'=====>> MAUVAISE REPONSE <<=====');
        end;
        goto menu;


end;    /* de la boucle while */

END DIGIS1;
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C INTEN
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme d'ecriture graphique en tons de gris dans le
C       systeme TRAPIX, d'une image provenent de la camera directement.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE INTEN (SPEC_FILE)
C
C
C       This program reads a part or the whole image buffer
C        from the specified disk file and write it into the
C        TRAPIX in intensity.
C
C       TRAPIX FORMAT
C
        IMPLICIT        INTEGER*2 (A-Z)
        CHARACTER*30    STRING,SPEC_FILE,AXE
        LOGICAL*4       SUCCESS
C
        PARAMETER       (NUM_PIX=512)
        INTEGER*4       IOS,KX,NORG,DELX,DELY
        INTEGER*2       FZI2(NUM_PIX)
        REAL*4          FACTZ
C
15      format(1A30)
20      format(8I4)
21      format(7I4)
25      format(F5.3)
C
C READ THE PARAMETERS
         OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='FORMATTED',IOSTAT=IOS)
        read(1,15) STRING
        read(1,15) axe
        read(1,20) dimx,dimy,INIX,INIY,FINX,FINY,X0,Y0
        read(1,21) WIND,CHANEL,FACTX,FACTY,SHIFT,INTERX,INTERY
        read(1,25) FACTZ
        close (unit=1,status='save')
C
        TRNSP = 0
        SUCCESS=INIT()
        IF(.NOT.SUCCESS)CALL FATAL_ERROR(SUCCESS,' ERROR : IN INIT')
C
        CALL FB2IDC( CHANEL,IDCNUM,CHAN,SCHN )  ! get user input
        DNTCRE  = WCHN(CHAN)                    ! set up write channel
        DNTCRE  = WSCH(SCHN)                    ! set up write sub-channel
```

```
/* Initialise le masque qui sert a identifier les mots de donnees */

        %REPLACE  MASQUE BY 'C000'b4 ;
        pt_valeur=addr(valeur_as_pointer);
        UNSPEC(para_rampe_ptr)='e000'b4;
        UNSPEC(para_pointer)='e100'b4;

        controle0 = '8109'b4 ;
        reset0 = '8009'b4;
        reset1 = reset0;
        controle1 = controle0;

IF IMODE=0 THEN GOTO MENU;

/* DEFINIE LA POSITION X0,Y0 ET X1,Y1 */

        TETA_INITIALE=PARAMETRES->CURSOR_POSITION(1);
        PHY_INITIALE=PARAMETRES->CURSOR_POSITION(2);
        TETA_FINAL=DONNEES->CURSOR_POSITION(1);
        PHY_FINAL=DONNEES->CURSOR_POSITION(2);
        PUT SKIP LIST ('DIMENSION DE L''IMAGE (#L,#C) ');
        GET LIST(NL,NC);
        INCREMENT_TETA=(TETA_FINAL-TETA_INITIALE)/(NC+1);
        NOMBRE_DE_LIGNE=NL;
        TETA_FINAL=TETA_INITIALE+INCREMENT_TETA*(NC+1);


/* DEMANDE DES PARAMETRES */
MENU:

put skip    list (clr,'                                    ');
put skip(3) list ('                          MENU POUR MODE RAMPE');

put skip(2) list ('1 - Position initiale (teta):        ',teta_initiale);
put skip list ('2 - Position initiale (phy):     ',phy_initiale);
put skip list ('3 - Position finale (teta):      ',teta_final);
put skip list ('4 - Position finale (phy):       ',phy_final);
put skip list ('5 - Vitesse horizontale (aller):         ',vitesse_teta_aller);
put skip list ('6 - Vitesse horizontale (retour):        ',vitesse_teta_retour);
put skip list ('7 - Vitesse verticale (aller):  ',vitesse_phy_aller);
put skip list ('8 - Increment horizontal:        ',Increment_teta);
put skip list ('9 - Nombre de lignes:    ',nombre_de_ligne);

put skip(2) list ('P - Envoi des parametres');
put skip    list ('L - Lecture du fichier de parametres');
put skip    list ('E - Ecriture vers le fichier de parametres');
put skip    list ('M - Retour au  menu principal');

/* ENCAS DE DIVISION PAR ZERO */
IF(increment_teta=0) then
        BEGIN;
        PUT SKIP(3) LIST('*** DIVISION PAR ZERO (Changer "8") ***');
        GOTO CHOIX(8);
        END;

/* Calcule le nombre de transferts */
```

```
curr_wrd = (teta_final - teta_initiale) /increment_teta;
curr_wrd = curr_wrd * nombre_de_ligne - nombre_de_ligne;

        DONNEES->DIM(1) = 1;
        DONNEES->DIM(2) = CURR_WRD;

 IF curr_wrd<0 THEN pcurr_wrd=dec(curr_wrd,6)+65535;
               ELSE pcurr_wrd=dec(curr_wrd,6);

put skip(3) list ('Le nombre de transferts est de ',pcurr_wrd);

PUT SKIP(2) LIST('Numero :  ');
GET LIST(numero);

        GOTO CHOIX(INDEX('123456789PLEM',numero));

CHOIX(0):
        DO;
        PUT SKIP(3) LIST('***** MAUVAISE REPONSE *****');
        end;
        goto menu;
CHOIX(1):
        DO;
        put list('Valeur : ');
        get  list (teta_initiale);
        call valmax1(teta_initiale);
        end;
        goto menu;
CHOIX(2):
        DO;
        put list('Valeur : ');
        get list (phy_initiale);
        call valmax1(phy_initiale);
        end;
        goto menu;
CHOIX(3):
        DO;
        put list('Valeur : ');
        get list (teta_final);
        call valmax1(teta_final);
        end;
        goto menu;
CHOIX(4):
        DO;
        put list('Valeur : ');
        get list (phy_final);
        call valmax1(phy_final);
        end;
        goto menu;
CHOIX(5):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_aller);
        call valmax1(vitesse_teta_aller);
        end;
        goto menu;
CHOIX(6):
```

```
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_retour);
        call valmax1(vitesse_teta_retour);
        end;
        goto menu;
CHOIX(7):
        DO;
        put list('Valeur : ');
        get list (vitesse_phy_aller);
        call valmax1(vitesse_phy_aller);
        end;
        goto menu;
CHOIX(8):
        DO;
        put list('Valeur : ');
        get list (increment_teta);
        call valmax2(increment_teta);
        end;
        goto menu;
CHOIX(9):
        DO;
        put list('Valeur : ');
        get list (nombre_de_ligne);
        call valmax2(nombre_de_ligne);
        end;
        goto menu;
CHOIX(10):
        DO;
        call envoie();
        end;
        goto menu;
CHOIX(11):
        DO;
        OPEN FILE (rampepar) UPDATE DIRECT ENV(F(128));
        READ FILE(rampepar) INTO(para_source) KEY(0) ;
        CLOSE FILE(rampepar);
        goto menu;
        end;
CHOIX(12):
        DO;
        OPEN FILE (rampepar) UPDATE DIRECT ENV(F(128));
        WRITE FILE(rampepar) FROM(para_source) KEYFROM(0) ;
        CLOSE FILE(rampepar);
        goto menu;
        end;
CHOIX(13):
        DO;
        END;

ENVOIE:
        PROCEDURE;

/* Initialise pour chaque mots de donnees les bits 14 et 15 */

        para7 = increment_teta * 16;
        para9 = nombre_de_ligne * 16;
```

```
        valeur_binary = teta_initiale;
        valeur_bit = valeur_bit ! masque;
        para1 = valeur_binary;

        valeur_binary = phy_initiale;
        valeur_bit = valeur_bit ! masque;
        para2 = valeur_binary;

        valeur_binary = teta_final;
        valeur_bit = valeur_bit ! masque;
        para3 = valeur_binary;

        valeur_binary = phy_final;
        valeur_bit = valeur_bit ! masque;
        para4  = valeur_binary;

        valeur_binary = vitesse_teta_aller;
        valeur_bit = valeur_bit ! masque;
        para5 = valeur_binary;

        valeur_binary = vitesse_teta_retour;
        valeur_bit = valeur_bit ! masque;
        para6 = valeur_binary;

        valeur_binary = para7;
        valeur_bit = valeur_bit ! masque;
        para7 = valeur_binary;

        valeur_binary = vitesse_phy_aller;
        valeur_bit = valeur_bit ! masque;
        para8 = valeur_binary;

        valeur_binary = para9;
        valeur_bit = valeur_bit ! masque;
        para9 = valeur_binary;


/* Envoie les parametres a la camera */

        parametres->dim(1)=1;
        parametres->dim(2)=13;
        call dma(3,parametres);
END ENVOIE;


VALMAX1:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>4095) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 4095 ***');
        PUT SKIP LIST(' Donnez une autre ');
        GOTO CHOIX(INDEX('123456789PM',numero));
        END;
END VALMAX1;
```

```
VALMAX2:
        PROCEDURE (VALEUR);
DCL     VALEUR              BINARY;
        IF(VALEUR>255) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 255 ***');
        PUT SKIP LIST(' Donnez une autre ');
        GOTO CHOIX(INDEX('123456789PM',numero));
        END;
END VALMAX2;


END RAMPE;
```

```
Esca10:
        PROCEDURE (PARAMETRES,DONNEES,IMODE);

        %INCLUDE 'STRUC.LIB';

DCL     dma entry (fixed bin(15),pointer);

DCL     (PARAMETRES,DONNEES)    POINTER;
DCL     IMODE                   BINARY;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES
            A ENVOYER POUR UNE IMAGE AVEC ESCALIER */
        para_pointer            pointer,
    1   para_source based(para_pointer),
        2 TETA_INITIALE         BINARY,
        2 PHY_INITIALE          BINARY,
        2 TETA_FINAL            BINARY,
        2 PHY_FINAL             BINARY,
        2 VITESSE_TETA_ALLER    BINARY,
        2 VITESSE_TETA_RETOUR   BINARY,
        2 INCREMENT_TETA        BINARY,
        2 INCREMENT_PHY         BINARY;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES MODIFIEES A ENVOYER */
        para_escalier_ptr    pointer,
    1   para_escalier based(para_escalier_ptr),
        2 controle0             bit(16),
        2 para1                 binary,
        2 para2                 binary,
        2 para3                 binary,
        2 para4                 binary,
        2 para5                 binary,
        2 para6                 binary,
        2 para7                 binary,
        2 para8                 binary,
        2 reset0                bit(16),
        2 reset1                bit(16),
        2 controle1             bit(16);

DCL     escalpar                file;

DCL
        curr_wrd        fixed bin(15),
        pcurr_wrd       fixed decimal(6),
        pt_valeur       pointer,
        valeur_bit      bit(16) based (pt_valeur) ,
        valeur_binary   BINARY  based (pt_valeur),
        valeur_as_pointer     POINTER,
        NL              BINARY,
        NC              BINARY;

DCL     PHY             FIXED BIN(15),
        CLR             CHARACTER(8),
        NUMERO          CHARACTER(1);

        CLR = '^[[2J ^[[H' :    /* EFFACE L'ECRAN */
```

```
/* Initialise le masque qui sert a identifier les mots de donnees */

        %REPLACE  MASQUE BY 'C000'b4 ;
        pt_valeur=addr(valeur_as_pointer);
        UNSPEC(para_escalier_ptr)='e000'b4;
        UNSPEC(para_pointer)='e200'b4;

        controle0 = '8148'b4 ;
        reset0 = '8048'b4;
        reset1 = reset0;
        controle1 = controle0;

IF IMODE=0 THEN GOTO MENU;

/* DEFINIE LA POSITION X0,Y0 ET X1,Y1 */

        TETA_INITIALE=PARAMETRES->CURSOR_POSITION(1);
        PHY_INITIALE= PARAMETRES->CURSOR_POSITION(2);
        TETA_FINAL =DONNEES->CURSOR_POSITION(1);
        PHY_FINAL  =DONNEES->CURSOR_POSITION(2);

        PUT SKIP LIST ('DIMENSION DE L''IMAGE (#L,#C)');
        GET LIST (NL,NC);
        IF NC=1 THEN INCREMENT_TETA=4095;
                ELSE     INCREMENT_TETA=(TETA_FINAL-TETA_INITIALE)/(NC-1);
        IF NL=1 THEN INCREMENT_PHY=4095;
                ELSE     INCREMENT_PHY =(PHY_FINAL-PHY_INITIALE)/(NL-1);

        IF INCREMENT_TETA=0 THEN INCREMENT_TETA=1;
        IF INCREMENT_PHY =0 THEN INCREMENT_PHY=1;
        TETA_FINAL=TETA_INITIALE+INCREMENT_TETA*(NC+1);
        PHY_FINAL=PHY_INITIALE+INCREMENT_PHY*NL;


/* DEMANDE DES PARAMETRES */
MENU:
put skip    list (clr,'                                                ');
put skip(3) list ('                          MENU POUR MODE ESCALIER');

put skip(2) list ('1 - Position initiale (teta):        ',teta_initiale);
put skip list ('2 - Position initiale (phy):     ',phy_initiale);
put skip list ('3 - Position finale (teta):      ',teta_final);
put skip list ('4 - Position finale (phy):       ',phy_final);
put skip list ('5 - Vitesse horizontale (aller):         ',vitesse_teta_aller);
put skip list ('6 - Vitesse horizontale (retour):        ',vitesse_teta_retour);
put skip list ('7 - Increment horizontal:        ',Increment_teta);
put skip list ('8 - Increment vertical:          ',Increment_phy);

put skip(2) list ('P - Envoi des parametres');
put skip    list ('L - Lecture du fichier de parametres');
put skip    list ('E - Ecriture vers le fichier de parametres');
put skip    list ('M - Retour au menu principal');

/* EN CAS DE DIVISION PAR ZERO */
IF (increment_teta=0) then
        BEGIN;
        PUT SKIP(3) LIST('*** DIVISION PAR ZERO (Changer "7") ***');
```

```
        GOTO CHOIX(7);
        END;
IF (increment_phy=0) then
        BEGIN;
        PUT SKIP(3) LIST('*** DIVISION PAR ZERO (Changer "8") ***');
        GOTO CHOIX(8);
        END;

        NC=((TETA_FINAL-TETA_INITIALE)/INCREMENT_TETA)-1;
        NL=(PHY_FINAL-PHY_INITIALE)/INCREMENT_PHY;
        CURR_WRD=NL*NC;

/* Calcule le nombre de transferts */
        IF curr_wrd<0 THEN pcurr_wrd=dec(curr_wrd,6)+65535;
              ELSE pcurr_wrd=dec(curr_wrd,6);

        DONNEES->DIM(1) = 1;
        DONNEES->DIM(2) = CURR_WRD;

put skip(3) list ('Le nombre de transferts est de ',pcurr_wrd);
PUT SKIP LIST ('#L= ',NL,' #C= ',NC);
PUT SKIP(2) LIST('Numero :  ');
GET LIST(numero);

        GOTO CHOIX(INDEX('12345678PLEM',numero));

CHOIX(0):
        DO;
        PUT SKIP(3) LIST('***** MAUVAISE REPONSE *****');
        end;
        goto menu;
CHOIX(1):
        DO;
        put list('Valeur : ');
        get  list (teta_initiale);
        call valmax1(teta_initiale);
        end;
        goto menu;
CHOIX(2):
        DO;
        put list('Valeur : ');
        get list (phy_initiale);
        call valmax1(phy_initiale);
        end;
        goto menu;
CHOIX(3):
        DO;
        put list('Valeur : ');
        get list (teta_final);
        call valmax1(teta_final);
        end;
        goto menu;
CHOIX(4):
        DO;
        put list('Valeur : ');
        get list (phy_final);
        call valmax1(phy_final);
```

```
        end;
        goto menu;
CHOIX(5):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_aller);
        call valmax1(vitesse_teta_aller);
        end;
        goto menu;
CHOIX(6):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_retour);
        call valmax1(vitesse_teta_retour);
        end;
        goto menu;
CHOIX(7):
        DO;
        put list('Valeur : ');
        get list (increment_teta);
        call valmax2(increment_teta);
        end;
        goto menu;
CHOIX(8):
        DO;
        put list('Valeur : ');
        get list (increment_phy);
        call valmax2(increment_phy);
        end;
        goto menu;
CHOIX(9):
        DO;
        CALL ENVOIE();
        end;
        goto menu;

CHOIX(10):
        DO;
        OPEN FILE (escalpar) UPDATE DIRECT ENV(F(128));
        READ FILE(escalpar) INTO(para_source) KEY(0) ;
        CLOSE FILE(escalpar);
        goto menu;
        end;
CHOIX(11):
        DO;
        OPEN FILE (escalpar) UPDATE DIRECT ENV(F(128));
        WRITE FILE(escalpar) FROM(para_source) KEYFROM(0) ;
        CLOSE FILE(escalpar);
        goto menu;
        end;
CHOIX(12):
        DO;
        END;

ENVOIE:
        PROCEDURE;
```

```
/* Initialise pour chaque mots de donnees les bits 14 et 15 */

        para7 = increment_teta * 16;
        para8 = increment_phy * 16;

        valeur_binary = teta_initiale;
        valeur_bit = valeur_bit ! masque;
        para1 = valeur_binary;

        valeur_binary = phy_initiale;
        valeur_bit = valeur_bit ! masque;
        para2 = valeur_binary;

        valeur_binary = teta_final;
        valeur_bit = valeur_bit ! masque;
        para3 = valeur_binary;

        valeur_binary = phy_final;
        valeur_bit = valeur_bit ! masque;
        para4  = valeur_binary;

        valeur_binary = vitesse_teta_aller;
        valeur_bit = valeur_bit ! masque;
        para5 = valeur_binary;

        valeur_binary = vitesse_teta_retour;
        valeur_bit = valeur_bit ! masque;
        para6 = valeur_binary;

        valeur_binary = para7;
        valeur_bit = valeur_bit ! masque;
        para7 = valeur_binary;

        valeur_binary = para8;
        valeur_bit = valeur_bit ! masque;
        para8 = valeur_binary;


/* Envoie les parametres a la camera */

        parametres->dim(1)=1;
        parametres->dim(2)=12;
        call dma(3,parametres);
END ENVOIE;


VALMAX1:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>4095) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 4095 ***');
        PUT SKIP LIST(' Donnez une autre ');
        GOTO CHOIX(INDEX('12345678PM',numero));
        END;
END VALMAX1;
```

```
VALMAX2:
        PROCEDURE (VALEUR);
DCL     VALEUR              BINARY;
        IF(VALEUR>255) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 255 ***');
        PUT SKIP LIST(' Donnez une autre ');
        GOTO CHOIX(INDEX('12345678PM',numero));
        END;
END VALMAX2;


END ESCAL0;
```

```
Ligne:
        PROCEDURE (PARAMETRES,DONNEES,IMODE);

        %INCLUDE 'STRUC.LIB';

DCL     dma entry (fixed bin(15),pointer);
DCL
        PARAMETRES                  POINTER,
        DONNEES                     POINTER,
        IMODE                       BINARY;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES
            A ENVOYER POUR UNE IMAGE AVEC LIGNE */
        para_pointer                pointer,
    1   para_source based(para_pointer),
        2 teta_initiale             binary,
        2 phy_initiale              binary,
        2 teta_final                binary,
        2 phy_final                 binary,
        2 vitesse_teta_aller        binary,
        2 vitesse_teta_retour       binary,
        2 increment                 binary,
        2 vitesse_phy_aller         binary;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES MODIFIEES A ENVOYER */
        para_ligne_ptr          pointer,
    1   para_ligne based(para_ligne_ptr),
        2 controle0                 bit(16),
        2 para1                     binary,
        2 para2                     binary,
        2 para3                     binary,
        2 para4                     binary,
        2 para5                     binary,
        2 para6                     binary,
        2 para7                     binary,
        2 para8                     binary,
        2 reset0                    bit(16),
        2 reset1                    bit(16),
        2 controle1                 bit(16);

DCL     lignepar                    file;

DCL
        curr_wrd            fixed bin(15),
        pcurr_wrd           fixed decimal(6),
        pt_valeur           pointer,
        valeur_bit          bit(16) based (pt_valeur) ,
        valeur_binary       BINARY  based (pt_valeur),
        valeur_as_pointer       POINTER;

DCL     CLR                 CHARACTER(1),
        NUMERO              CHARACTER(1);

        CLR = '^[[2J ^[[H' ;     /* EFFACE L'ECRAN */

DCL     ANGLE               FLOAT BINARY,
        NC                  BINARY;
```

```
/* Initialise le masque qui sert a identifier les mots de donnees */

        %REPLACE  MASQUE BY 'C000'b4 ;
        pt_valeur=addr(valeur_as_pointer);
        UNSPEC(para_ligne_ptr)='e000'b4;
        UNSPEC(para_pointer)='e300'b4;

IF IMODE=0 THEN GOTO MENU;

/* DEFINIE LA POSITION X0,Y0 ET X1,Y1     */
        TETA_INITIALE=PARAMETRES->CURSOR_POSITION(1);
        PHY_INITIALE=PARAMETRES->CURSOR_POSITION(2);
        TETA_FINAL=DONNEES->CURSOR_POSITION(1);
        PHY_FINAL=DONNEES->CURSOR_POSITION(2);
        PUT SKIP LIST ('NOMBRE DE POINT #C ');
        GET LIST (NC);
 IF NC<=1 THEN INCREMENT=255;
 ELSE   DO;
        ANGLE=FLOAT(PHY_FINAL-PHY_INITIALE)/FLOAT(TETA_FINAL-TETA_INITIALE);
        ANGLE=ATAN(ANGLE)*180.0/3.14159;
                IF (ANGLE<=45.0)
                THEN DO;
                        INCREMENT=(TETA_FINAL-TETA_INITIALE)/(NC-1);
                        TETA_FINAL=TETA_INITIALE+INCREMENT*(NC+1);
                     END;
                ELSE DO;
                        INCREMENT=(PHY_FINAL-PHY_INITIALE)/(NC-1);
                        PHY_FINAL=PHY_INITIALE+INCREMENT*(NC+1);
                     END;
        END;
/* DEMANDE DES PARAMETRES */
MENU:

put skip    list (clr,'                                        ');
put skip(3) list ('                          MENU POUR MODE LIGNE');

put skip(2) list ('1 - Position initiale (teta):          ',teta_initiale);
put skip list ('2 - Position initiale (phy):     ',phy_initiale);
put skip list ('3 - Position finale (teta):      ',teta_final);
put skip list ('4 - Position finale (phy):       ',phy_final);
put skip list ('5 - Vitesse horizontale (aller):          ',vitesse_teta_aller);
put skip list ('6 - Vitesse horizontale (retour):         ',vitesse_teta_retour);
put skip list ('7 - Vitesse verticale (aller):  ',vitesse_phy_aller);
put skip list ('8 - Increment (horizontal ou vertical): ',Increment);

put skip(2) list ('P - Envoi des parametres');
put skip    list ('L - Lecture du fichier de parametres');
put skip    list ('E - Ecriture vers le fichier de parametres');
put skip    list ('M - Retour au menu principal');

/* DETERMINATION DE L'ANGLE ET CALCUL DU NOMBRE DE TRANSFERTS */

IF(teta_initiale-teta_final = 0) then
        begin;
        PUT SKIP LIST('*** DIVISION PAR ZERO ***(Changer "1")');
        goto choix(1);
```

```
        end;

IF (ANGLE<=45.0) THEN
        DO;
        controle0 = '8118'b4 ;
        reset0 = '8018'b4;
        reset1 = reset0;
        controle1 = controle0;
        IF(Increment=0) then
                BEGIN;
                PUT SKIP(3) LIST('*** DIVISION PAR ZERO (Changer "8") ***');
                GOTO CHOIX(8);
                END;
        curr_wrd = (teta_final - teta_initiale) / increment - 1;
        END;
ELSE
        DO;
        controle0 = '8128'b4 ;
        reset0 = '8028'b4;
        reset1 = reset0;
        controle1 = controle0;
        IF(Increment=0) then
                BEGIN;
                PUT SKIP(3) LIST('*** DIVISION PAR ZERO (Changer "8") ***');
                GOTO CHOIX(8);
                END;
        curr_wrd = (phy_final - phy_initiale) / increment - 1;
        END;

        DONNEES->DIM(1) = 1;
        DONNEES->DIM(2) = CURR_WRD;
 IF CURR_WRD<0 THEN pcurr_wrd=dec(curr_wrd,6)+65535;
               ELSE pcurr_wrd=dec(curr_wrd,6);

put skip(3) list ('Le nombre de transferts est de ',pcurr_wrd);

PUT SKIP(2) LIST('Numero :  ');
GET LIST(numero);

        GOTO CHOIX(INDEX('12345678PLEM',numero));

CHOIX(1):
        DO;
        put list('Valeur : ');
        get  list (teta_initiale);
        call valmax1(teta_initiale);
        end;
        goto menu;
CHOIX(2):
        DO;
        put list('Valeur : ');
        get list (phy_initiale);
        call valmax1(phy_initiale);
        end;
        goto menu;
CHOIX(3):
        DO;
```

```
        put list('Valeur : ');
        get list (teta_final);
        call valmax1(teta_final);
        end;
        goto menu;
CHOIX(4):
        DO;
        put list('Valeur : ');
        get list (phy_final);
        call valmax1(phy_final);
        end;
        goto menu;
CHOIX(5):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_aller);
        call valmax1(vitesse_teta_aller);
        end;
        goto menu;
CHOIX(6):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_retour);
        call valmax1(vitesse_teta_retour);
        end;
        goto menu;
CHOIX(7):
        DO;
        put list('Valeur : ');
        get list (vitesse_phy_aller);
        call valmax1(vitesse_phy_aller);
        end;
        goto menu;
CHOIX(8):
        DO;
        put list('Valeur : ');
        get list (increment);
        call valmax2(increment);
        end;
        goto menu;
CHOIX(0):
        DO;
        PUT SKIP(3) LIST('***** MAUVAISE REPONSE *****');
        end;
        goto menu;
CHOIX(9):
        DO;

/* Initialise pour chaque mots de donnees les bits 14 et 15 */

        para7 = increment * 16;

        valeur_binary = teta_initiale;
        valeur_bit = valeur_bit | masque;
        para1 = valeur_binary;

        valeur_binary = phy_initiale;
```

```
        valeur_bit = valeur_bit ! masque;
        para2 = valeur_binary;

        valeur_binary = teta_final;
        valeur_bit = valeur_bit ! masque;
        para3 = valeur_binary;

        valeur_binary = phy_final;
        valeur_bit = valeur_bit ! masque;
        para4  = valeur_binary;

        valeur_binary = vitesse_teta_aller;
        valeur_bit = valeur_bit ! masque;
        para5 = valeur_binary;

        valeur_binary = vitesse_teta_retour;
        valeur_bit = valeur_bit ! masque;
        para6 = valeur_binary;

        valeur_binary = para7;
        valeur_bit = valeur_bit ! masque;
        para7 = valeur_binary;

        valeur_binary = vitesse_phy_aller;
        valeur_bit = valeur_bit ! masque;
        para8 = valeur_binary;


/* Envoie les parametres a la camera */

        parametres->dim(1)=1;
        parametres->dim(2)=12;
        call dma(3,parametres);

PUT SKIP LIST('L''angle entre la ligne et l''horizontale');
PUT SKIP EDIT('est de ',angle,' degres ')
       (A(9),F(5,2),A(10));

        end;
        goto menu;
CHOIX(10):
        DO;
        OPEN FILE (lignepar) UPDATE DIRECT ENV(F(128));
        READ FILE(lignepar) INTO(para_source) KEY(0) ;
        CLOSE FILE (lignepar);
        goto menu;
        end;
CHOIX(11):
        DO;
        OPEN FILE (lignepar) UPDATE DIRECT ENV(F(128));
        WRITE FILE(lignepar) FROM(para_source) KEYFROM(0) ;
        CLOSE FILE (lignepar);
        goto menu;
        end;
CHOIX(12):
        DO;
        END;
```

```
VALMAX1:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>4095) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 4095 ***');
        PUT SKIP LIST(' Donnez une autre');
        GOTO CHOIX(INDEX('12345678PM',numero));
        END;
END VALMAX1;


VALMAX2:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>255) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 255 ***');
        PUT SKIP LIST(' Donnez une autre');
        GOTO CHOIX(INDEX('12345678PM',numero));
        END;
END VALMAX2;


END LIGNE;
```

```
Point:
        PROCEDURE (PARAMETRES,DONNEES,IMODE) ;

        %INCLUDE 'STRUC.LIB';

DCL     dma entry(fixed bin(15),pointer);

DCL     (PARAMETRES,DONNEES)    POINTER;
DCL     IMODE                   BINARY ;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES
            A ENVOYER POUR UNE IMAGE AVEC POINT */
        para_pointer            pointer,
    1   para_source based(para_pointer),
        2 teta                  binary,
        2 phy                   binary,
        2 precision_teta        binary,
        2 precision_phy         binary;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES MODIFIEES A ENVOYER */
        para_point_ptr          pointer,
    1   para_point based(para_point_ptr),
        2 controle0             bit(16),
        2 para1                 binary,
        2 para2                 binary,
        2 para3                 binary,
        2 para4                 binary,
        2 reset0                bit(16),
        2 reset1                bit(16),
        2 controle1             bit(16);

DCL     pointpar                file;

DCL
        curr_wrd        fixed bin(15),
        pt_valeur       pointer,
        valeur_bit      bit(16) based (pt_valeur) ,
        valeur_binary   BINARY  based (pt_valeur),
        valeur_as_pointer     POINTER;

DCL     CLR             CHARACTER(8),
        NUMERO          CHARACTER(1);

        CLR = '^[[2J ^[[H' ;    /* EFFACE L'ECRAN */

/* Initialise le masque qui sert a identifier les mots de donnees */

        %REPLACE  MASQUE BY 'C000'b4 ;
        pt_valeur=addr(valeur_as_pointer);
        UNSPEC(para_point_ptr)='e000'b4;
        UNSPEC(para_pointer)='e400'b4;

        controle0 = '8134'b4 ;
        reset0 = '8034'b4;
        reset1 = reset0;
        controle1 = controle0;
```

```
        DONNEES->DIM(1) = 1;
        DONNEES->DIM(2) = 1;

   IF IMODE=0 THEN GOTO MENU;

        TETA=PARAMETRES->CURSOR_POSITION(1);
        PHY =PARAMETRES->CURSOR_POSITION(2);

/* DEMANDE DES PARAMETRES */
MENU:

put skip    list (clr,'                                          ');
put skip(3) list ('                              MENU POUR MODE POINT');

put skip(2) list ('1 - Position initiale (teta):          ',teta);
put skip list ('2 - Position initiale (phy):     ',phy);
put skip list ('3 - Precision (teta):    ',precision_teta);
put skip list ('4 - Precision (phy):     ',precision_phy);

put skip(2) list ('P - Envoi des parametres');
put skip    list ('L - Lecture du fichier de parametres');
put skip    list ('E - Ecriture vers le fichier de parametres');
put skip    list ('M - Retour au menu principal');

curr_wrd = 1;
put skip(3) list ('Le nombre de transferts est de ',curr_wrd);

PUT SKIP(2) LIST('Numero :  ');
GET LIST(numero);

        GOTO CHOIX(INDEX('1234PLEM',numero));

CHOIX(0):
        DO;
        PUT SKIP(3) LIST('***** MAUVAISE REPONSE *****');
        end;
        goto menu;
CHOIX(1):
        DO;
        put list('Valeur : ');
        get  list (teta);
        call valmax1(teta);
        end;
        goto menu;
CHOIX(2):
        DO;
        put list('Valeur : ');
        get list (phy);
        call valmax1(phy);
        end;
        goto menu;
CHOIX(3):
        DO;
        put list('Valeur : ');
        get list (precision_teta);
        call valmax2(precision_teta);
        end;
```

```
        goto menu;
CHOIX(4):
        DO;
        put list('Valeur : ');
        get list (precision_phy);
        call valmax2(precision_phy);
        end;
        goto menu;
CHOIX(5):
        DO;
        CALL ENVOIE();
        end;
        goto menu;

CHOIX(6):
        DO;
        OPEN FILE (pointpar) UPDATE DIRECT ENV(F(128));
        READ FILE(pointpar) INTO(para_source) KEY(0) ;
        CLOSE FILE (pointpar);
        goto menu;
        end;
CHOIX(7):
        DO;
        OPEN FILE (pointpar) UPDATE DIRECT ENV(F(128));
        WRITE FILE(pointpar) FROM(para_source) KEYFROM(0) ;
        CLOSE FILE (pointpar);
        goto menu;
        end;
CHOIX(8):
        DO;
        END;

ENVOIE:
        PROCEDURE;
/* Initialise pour chaque mots de donnees les bits 14 et 15 */

        para3 = precision_teta * 16;
        para4 = precision_phy * 16;

        valeur_binary = teta;
        valeur_bit = valeur_bit | masque;
        para1 = valeur_binary;

        valeur_binary = phy;
        valeur_bit = valeur_bit | masque;
        para2 = valeur_binary;

        valeur_binary = para3;
        valeur_bit = valeur_bit | masque;
        para3 = valeur_binary;

        valeur_binary = para4;
        valeur_bit = valeur_bit | masque;
        para4  = valeur_binary;

        DONNEES->DIM(1) = 1;
        DONNEES->DIM(2) = 1;
```

```
/* Envoie les parametres a la camera */

        parametres->dim(1)=1;
        parametres->dim(2)=8;
        call dma(3,parametres);

END ENVOIE;


VALMAX1:
        PROCEDURE (VALEUR);

DCL     VALEUR          BINARY;
        IF(VALEUR>4095) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 4095 ***');
        PUT SKIP LIST('Donnez une autre');
        GOTO CHOIX(INDEX('1234PM',numero));
        END;

END VALMAX1;


VALMAX2:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>255) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 255 ***');
        PUT SKIP LIST('Donnez une autre');
        GOTO CHOIX(INDEX('1234PM',numero));
        END;

END VALMAX2;


END point;
```

```
ACQUI1:
                PROCEDURE (DONNEES,MOT_ACQUIS);

/* ROUTINE D'ACQUISITION DES COORDONNEES D'UN SOLIDE V 1.2*/

        %INCLUDE 'STRUC.LIB';

/* STRUCTURE DU DPS1 */
        %REPLACE  PORT_A  BY  '08'B4,  /*PORT PARALLELE */
                  PORT_B  BY  '09'B4;

DCL             (ON,OFF)          BIT(8);

DCL             DONNEES           POINTER,
                ADDR_PTR          POINTER,
                MOT_ACQUIS        POINTER;

DCL             ALPHA_VARIABLE    BINARY,

                I                 BINARY,
                DEP               DECIMAL,
                ADDR_DEC          DECIMAL,
                NC                BINARY;

DCL             POUTPUT ENTRY(BIT(8),BIT(8)),
                CONVERT ENTRY(POINTER,POINTER,DECIMAL,BINARY),
                DMA     ENTRY(FIXED BIN(15),POINTER),
                REMAP   ENTRY(BINARY,POINTER);

/*STRUCTURE DES PARAMETERS D'ACQUISITION */

DCL     ACQ_PARMS_PTR             POINTER,

        1  ACQ_PARMS              BASED(ACQ_PARMS_PTR),
           2 PARM1                BINARY,
           2 PARM2                BINARY,
           2 PARM3                BINARY,
           2 PARM4                BINARY,
           2 PARM5                BINARY,
           2 PARM6                BINARY;

DCL        ALPHA_INITIAL          BINARY,
           ALPHA_FINAL            BINARY,
           ALPHA_DEL              BINARY,
           NB_POINT               BINARY,
           NB_PASSES_INT          BINARY,
           POS_PAGE               BINARY,
           DEL_VAL                BINARY;

        UNSPEC(ACQ_PARMS_PTR)='E200'B4;

/* INITIALISATION DES VARIABLES */

                POS_PAGE=1;
                ALPHA_INITIAL=PARM1;
                ALPHA_FINAL=PARM2;
                ALPHA_DEL=PARM3;
```

```
                NB_POINT=PARM4;
                NB_PASSES_INT=PARM5;
                DEL_VAL=PARM6;
                NC=(ALPHA_FINAL-ALPHA_INITIAL)/ALPHA_DEL;
                ON ='01'B4;
                OFF='00'B4;
                I=1;
                ALPHA_VARIABLE=ALPHA_INITIAL;
                DEP=DEC(DONNEES->DIM(2)*2);
                ADDR_DEC=131072;/* ( 20000H ) */

/*INITIALISE LE PORT (B) A 1*/

                CALL POUTPUT(PORT_B,OFF);

/* DEBUTE L'ACQUISITION D'UNE LIGNE DANS LA MEMOIRE ETENDUE*/

                CALL CONVERT(DONNEES,ADDR_PTR,ADDR_DEC,POS_PAGE);

BOUCLE:
                PUT SKIP(0) LIST ('ANGLE = ',ALPHA_VARIABLE);
                CALL ACQ_DEBUT(DONNEES,MOT_ACQUIS);

/*DEBUTE LA ROTATION DE LA TABLE */

        IF ALPHA_VARIABLE<ALPHA_FINAL THEN DO;

                CALL POUTPUT(PORT_B,ON);
                CALL DELAI(10);
                CALL POUTPUT(PORT_B,OFF);
        END;

/*CALCUL DE L'ADRESSE DU PROCHAIN TRANSFERT */

                ADDR_DEC=ADDR_DEC+DEP;
                CALL CONVERT(DONNEES,ADDR_PTR,ADDR_DEC,POS_PAGE);

/*DELAI SUPLEMENTAIRE POUR LA STABILISATION DE LA TABLE TOURNANTE*/

                CALL DELAI(DEL_VAL);
                CALL DELAI(DEL_VAL);

/*      FIN?      */

                ALPHA_VARIABLE=ALPHA_VARIABLE+ALPHA_DEL;
                I=I+1;

        IF ALPHA_VARIABLE<=ALPHA_FINAL THEN GOTO BOUCLE;

                RETURN;


/* DEBUTE L'ACQUISITION D'UNE LIGNE ET TRANSFERT LES DONNEES DANS
   LA MEMOIRE ETENDUE SITUEE A '00021000'X */

ACQ_DEBUT:
                PROCEDURE (DONNEES,MOT_ACQUIS);
```

```
        DCL     (DONNEES,MOT_ACQUIS)    POINTER;

                CALL REMAP (1,MOT_ACQUIS);
                CALL DMA   (3,MOT_ACQUIS);
                CALL REMAP (1,DONNEES);
                CALL DMA   (2,DONNEES);

END ACQ_DEBUT;


/* DELAI DE STABILISATION MECANIQUE DE LA TABLE TOURNANTE */

DELAI:
                PROCEDURE (DEL_VAL);

        DCL     DEL_VAL         BINARY,
                (TEMP,TEMP1)    DECIMAL;

                DO  TEMP=1 TO DEC(DEL_VAL);
                TEMP1=TEMP1+1;
                END;
END  DELAI;

END  ACQUI1;
```

```
SETXY:
        PROCEDURE (PARAMETRES,DONNEES) ;

        %INCLUDE 'STRUC.LIB';

DCL     dma entry(fixed bin(15),pointer);

DCL     (PARAMETRES,DONNEES)    POINTER;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES
            A ENVOYER POUR UNE IMAGE AVEC POINT */
        para_pointer            pointer,
    1   para_source based(para_pointer),
        2 teta(2)               binary,
        2 phy(2)                binary,
        2 precision_teta        binary,
        2 precision_phy         binary;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES MODIFIEES A ENVOYER */
        para_point_ptr          pointer,
    1   para_point based(para_point_ptr),
        2 controle0             bit(16),
        2 para1                 binary,
        2 para2                 binary,
        2 para3                 binary,
        2 para4                 binary,
        2 reset0                bit(16),
        2 reset1                bit(16),
        2 controle1             bit(16);

DCL
        pt_valeur       pointer,
        valeur_bit      bit(16) based (pt_valeur) ,
        valeur_binary   BINARY  based (pt_valeur),
        valeur_as_pointer     POINTER;
DCL     ESC             CHARACTER(1),
        (HOME,CLR)      CHARACTER(8);

/* DEFINIE LES SEQUENCES DE CONTROLES POUR UN TERMINAL VT102 */
        ESC='^[';
        HOME=ESC||'[H'||ESC||'[K';
        CLR= ESC||'[2J';

/* Initialise le masque qui sert a identifier les mots de donnees */

        %REPLACE  MASQUE BY 'C000'b4 ;
        pt_valeur=addr(valeur_as_pointer);
        UNSPEC(para_point_ptr)='e000'b4;
        UNSPEC(para_pointer)='e500'b4;

        controle0 = '8134'b4 ;
        reset0 = '8034'b4;
        reset1 = reset0;
        controle1 = controle0;
        PRECISION_PHY=1;
        PRECISION_TETA=1;
        DONNEES->DIM(1) = 1;
```

```
        DONNEES->DIM(2) = 1;
        PARAMETRES->DIM(1)=1;
        PARAMETRES->DIM(2)=8;

DECLARE (INITSTAT,TETA1,PHY1,STEP) BINARY;

        STEP=10;

%REPLACE T_DATA_A   BY '00'B4;
%REPLACE T_STATUS_A BY '01'B4;

DCL  PINPUT ENTRY (BIT(8)) RETURNS(BIT(8));
DCL  POUTPUT ENTRY (BIT(8),BIT(8)) EXTERNAL;
DCL  STATUS          BIT(8),
     COMMAND         BINARY,
     COMMAND_BYTE    BIT(8)       BASED (COMMAND_PTR),
     COMMAND_CHAR    CHARACTER(1) BASED (COMMAND_PTR),
     COMMAND_PTR     POINTER;

     COMMAND_PTR=ADDR(COMMAND);

/* LECTURE DES PARAMETRES INITIAUX */

        PUT SKIP LIST (CLR,HOME);
        PUT SKIP LIST ('TETA INITIALE = ');
        GET LIST (TETA1);
 IF TETA1=-1 THEN TETA1=TETA(1);
 IF TETA1=-2 THEN TETA1=TETA(2);
        PUT SKIP LIST ('PHY  INITIALE = ');
        GET LIST (PHY1);
 IF PHY1=-1 THEN PHY1=PHY(1);
 IF PHY1=-2 THEN PHY1=PHY(2);

/* CODAGE DES DONNEES DE PRECISION */
        PARA3=PRECISION_TETA*16;
        PARA4=PRECISION_PHY *16;
        VALEUR_BINARY=PARA3;
        VALEUR_BIT=VALEUR_BIT!MASQUE;
        PARA3=VALEUR_BINARY;
        VALEUR_BINARY=PARA4;
        VALEUR_BIT=VALEUR_BIT!MASQUE;
        PARA4=VALEUR_BINARY;
        INITSTAT=0;
GOTO TRANS;

/* MENU DE L''EDITEUR DE POINT */

CHOIX(5):
          DO;
          PUT LIST(CLR,HOME);
          PUT SKIP (3) LIST ('MENU DE L''EDITEUR DE POINT');
          PUT SKIP     LIST ('(2) CURSEUR VERS LE BAS');
          PUT SKIP     LIST ('(4) CURSEUR VERS LA GAUCHE');
          PUT SKIP     LIST ('(6) CURSEUR VERS LA DROITE');
          PUT SKIP     LIST ('(8) CURSEUR VERS LE HAUT ');
          PUT SKIP     LIST ('(M) MENU ');
          PUT SKIP     LIST ('(0) DEFINIE X0,Y0');
```

```
          PUT SKIP      LIST ('(1) DEFINIE X1,Y1');
          PUT SKIP      LIST ('(D) PAS DU DEPLACEMENT');
          PUT SKIP      LIST ('(S) ARRET');
          PUT SKIP      LIST (HOME) ;
  IF INITSTAT=0 THEN PUT LIST ('TETA =',TETA1,' PHY=',PHY1);
          END;

BOUC:
          STATUS=0;
          INITSTAT=1;
     DO WHILE (STATUS=0);
          STATUS=PINPUT(T_STATUS_A)&'02'B4;
     END;
          COMMAND_BYTE=PINPUT(T_DATA_A);
          COMMAND_BYTE=COMMAND_BYTE & '7F'B4;

      GOTO CHOIX(INDEX('8462MOD1S',COMMAND_CHAR)) ;

CHOIX(0):
       DO;
          PUT SKIP LIST (HOME,' CARACTERE (M) POUR LISTE DES CONTROLES ->');
          PUT LIST (COMMAND_CHAR);
       END;
 GOTO BOUC;
CHOIX(1):
         /*MOUVEMENT VERS LE BAS */

       DO;
          PHY1=PHY1-STEP;
          IF PHY1<=0 THEN  PHY1=PHY1+STEP;
       END;
GOTO TRANS;

CHOIX(2):
         /*MOUVEMENT VERS LA GAUCHE */

       DO;
          TETA1=TETA1-STEP;
          IF TETA1<=0 THEN TETA1=TETA1+STEP;
       END;
GOTO TRANS;

CHOIX(3):
         /* MOUVEMENT VERS LA DROITE */
       DO;
          TETA1=TETA1+STEP;
          IF TETA1>4095 THEN TETA1=TETA1-STEP;
       END;
GOTO TRANS;

CHOIX(4):
         /*MOUVEMENT VERS LE HAUT */
       DO;
          PHY1=PHY1+STEP;
          IF PHY1>4095 THEN PHY1=PHY1-STEP;
       END;
```

```
TRANS:
       DO;
        PUT SKIP LIST(HOME,'TETA=',TETA1,' PHY=',PHY1);
        valeur_binary = teta1;
        valeur_bit = valeur_bit | masque;
        para1 = valeur_binary;

        valeur_binary = phy1;
        valeur_bit = valeur_bit | masque;
        para2 = valeur_binary;

/* Envoie les parametres a la camera */

        call dma(3,parametres);

        end;
   IF INITSTAT=1 THEN GOTO BOUC; ELSE GOTO CHOIX(5);

CHOIX(6):
         /* DEFINIE LA VALEUR X0,Y0 */
       DO;
          TETA(1)=TETA1;
          PHY(1) =PHY1;
          PARAMETRES->CURSOR_POSITION(1)=TETA1;
          PARAMETRES->CURSOR_POSITION(2)=PHY1;
          PUT SKIP  LIST (HOME,' TETA0= ',TETA(1),' PHY0= ',PHY(1));
       END;
 GOTO  BOUC;

CHOIX(7):
         /* DEFINIE LE PAS DU CURSEUR */
       DO;
          COMMAND_BYTE=PINPUT(T_DATA_A);
          PUT SKIP  LIST (HOME,' VALEUR DU PAS :');
          GET LIST (STEP);
       END;
          PUT SKIP LIST (HOME,'TETA=',TETA1,' PHY=',PHY1);
  GOTO  BOUC;

CHOIX(8):
         DO;
            PHY(2)=PHY1;
            TETA(2)=TETA1;
            DONNEES->CURSOR_POSITION(1)=TETA1;
            DONNEES->CURSOR_POSITION(2)=PHY1;
            PUT SKIP LIST (HOME,' TETA1= ',TETA(2),' PHY1= ',PHY(2));
         END;
GOTO BOUC;

CHOIX(9):
         DO;
            PUT LIST (CLR);
         END;

END SETXY;
```

```
DEFACQ:
        PROCEDURE (PARAMETRES,DONNEES,IMODE) ;

        %INCLUDE 'STRUC.LIB';

DCL     IMODE           BINARY;

DCL     dma entry (fixed bin(15),pointer);

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES
            A ENVOYER POUR UNE IMAGE AVEC RAMPE. */
        para_pointer            pointer,
    1   para_source based(para_pointer),
        2 teta_initiale         binary,
        2 phy_initiale          binary,
        2 teta_final            binary,
        2 phy_final             binary,
        2 vitesse_teta_aller    binary,
        2 vitesse_teta_retour   binary,
        2 increment_teta        binary,
        2 vitesse_phy_aller     binary;/*vitesse mise a zero */

        VITESSE_PHY_ALLER=1000;

DECLARE /* STRUCTURE DES PARAMETRES D'ACQUISITION */

        ACQ_PARMS_PTR           POINTER,

    1   ACQ_PARMS               BASED(ACQ_PARMS_PTR),
        2 ALPHA_INITIAL         BINARY,
        2 ALPHA_FINAL           BINARY,
        2 ALPHA_DEL             BINARY,
        2 NB_POINTS             BINARY,
        2 NB_PASSES_INT         BINARY,
        2 DEL_VAL               BINARY,
        2 TRANS                 BINARY;

        UNSPEC(ACQ_PARMS_PTR)='e200'B4;

DECLARE  /* STRUCTURE CONTENANT LES PARAMETRES MODIFIEES A ENVOYER */
        para_rampe_ptr          pointer,
    1   para_rampe based(para_rampe_ptr),
        2 controle0             bit(16),
        2 para1                 binary,
        2 para2                 binary,
        2 para3                 binary,
        2 para4                 binary,
        2 para5                 binary,
        2 para6                 binary,
        2 para7                 binary,
        2 para8                 binary,
        2 para9                 binary,
        2 reset0                bit(16),
        2 reset1                bit(16),
        2 controle1             bit(16);

DCL     acqpar          file;
```

```
DCL       i                          BINARY,
          (PARAMETRES,DONNEES)       POINTER;

DCL
          curr_wrd          fixed bin(15),
          pcurr_wrd         fixed decimal(6),
          pt_valeur         pointer,
          valeur_bit        bit(16) based (pt_valeur) ,
          valeur_binary     BINARY  based (pt_valeur),
          valeur_as_pointer      POINTER;

DCL       NUMERO            CHARACTER(1);

DCL       CLR               CHARACTER(8);

          CLR='^[[2J ^[[H';

/* Initialise le masque qui sert a identifier les mots de donnees */

          %REPLACE  MASQUE BY 'C000'b4 ;
          pt_valeur=addr(valeur_as_pointer);
          UNSPEC(para_rampe_ptr)='e000'b4;
          UNSPEC(para_pointer)='e100'b4;

          controle0 = '8109'b4 ;
          reset0 = '8009'b4;
          reset1 = reset0;
          controle1 = controle0;

IF IMODE=0 THEN;
          DO;
          OPEN FILE (acqpar) UPDATE DIRECT ENV(F(128));
          READ FILE(acqpar) INTO(para_source) KEY(0) ;
          READ FILE(acqpar) INTO(acq_parms) KEY(1) ;
          CLOSE FILE(acqpar);
          goto menu;
          end;

/* DEFINIE LA POSITION X0,Y0 ET X1,Y1 */

          TETA_INITIALE=PARAMETRES->CURSOR_POSITION(1);
          PHY_INITIALE=PARAMETRES->CURSOR_POSITION(2);
          TETA_FINAL=DONNEES->CURSOR_POSITION(1);
          PHY_FINAL=DONNEES->CURSOR_POSITION(2);
          PUT SKIP LIST ('Nombre de points dans la ligne ');
          GET LIST(NB_POINTS);
          PUT SKIP LIST ('Nombre de passe d''integration ');
          GET LIST(NB_PASSES_INT);
          PUT SKIP LIST ('Angle initial (Deg.) ');
          GET LIST (ALPHA_INITIAL);
          PUT SKIP LIST ('Angle final   (Deg.) ');
          GET LIST (ALPHA_FINAL);
          PUT SKIP LIST ('Increment angulaire de la table tournante (deg.) ');
          GET SKIP LIST(ALPHA_DEL);
          PUT SKIP LIST ('Delai de   ilisation de la table tournante   ');
          GET LIST (DEL_VAL);
```

```
        INCREMENT_TETA=(TETA_FINAL-TETA_INITIALE)/(NB_POINTS+1);
        TETA_FINAL=TETA_INITIALE+INCREMENT_TETA*(NB_POINTS);


/* DEMANDE DES PARAMETRES */
MENU:
PUT LIST (CLR);
put skip list ('                              MENU POUR DEFINIR L''ACQUISITION');

put skip(2) list ('1 - Position initiale (teta):        ',teta_initiale);
put skip list ('2 - Position initiale (phy):     ',phy_initiale);
put skip list ('3 - Position finale (teta):      ',teta_final);
put skip list ('4 - Position finale (phy):       ',phy_final);
put skip list ('5 - Vitesse horizontale (aller):         ',vitesse_teta_aller);
put skip list ('6 - Vitesse horizontale (retour):        ',vitesse_teta_retour);
put skip list ('7 - Alpha initial  ',alpha_initial);
put skip list ('8 - Alpha final    ',alpha_final);
put skip list ('9 - increment de la table tournante :  ',ALPHA_DEL);
put skip list ('A - Increment horizontal:        ',Increment_teta);
put skip list ('B - Nombre de passes d''integration:     ',NB_PASSES_INT);
put skip list ('C - Delai de stabilisation de la table  ',del_val);
put skip list ('D - Ajustement du nombre de transfert  ',trans);
put skip list ('P - Envoi des parametres');
put skip list ('L - Lecture du fichier de parametres');
put skip list ('E - Ecriture vers le fichier de parametres');
put skip list ('M - Retour au  menu principal');

/* Calcule le nombre de transferts pour un angle */
IF(increment_teta=0) then
        BEGIN;
        PUT SKIP(3) LIST('*** DIVISION PAR ZERO (Changer "8") ***');
        GOTO CHOIX(8);
        END;
curr_wrd = (teta_final - teta_initiale) /increment_teta;
curr_wrd = (curr_wrd * NB_PASSES_INT) - trans * NB_PASSES_INT ;

        DONNEES->DIM(1) = 1;
        DONNEES->DIM(2) = CURR_WRD;

 IF curr_wrd<0 THEN pcurr_wrd=dec(curr_wrd,6)+65535;
               ELSE pcurr_wrd=dec(curr_wrd,6);

put skip(3) list ('Le nombre de transferts pour un angle est de ',pcurr_wrd);

PUT SKIP(2) LIST('Numero :  ');
GET LIST(numero);

        GOTO CHOIX(INDEX('123456789ABCPLEMD',numero));

CHOIX(1):
        DO;
        put list('Valeur : ');
        get  list (teta_initiale);
        call valmax1(teta_initiale);
        end;
        goto menu;
CHOIX(2):
```

```
        DO;
        put list('Valeur : ');
        get list (phy_initiale);
        call valmax1(phy_initiale);
        end;
        goto menu;
CHOIX(3):
        DO;
        put list('Valeur : ');
        get list (teta_final);
        call valmax1(teta_final);
        end;
        goto menu;
CHOIX(4):
        DO;
        put list('Valeur : ');
        get list (phy_final);
        call valmax1(phy_final);
        end;
        goto menu;
CHOIX(5):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_aller);
        call valmax1(vitesse_teta_aller);
        end;
        goto menu;
CHOIX(6):
        DO;
        put list('Valeur : ');
        get list (vitesse_teta_retour);
        call valmax1(vitesse_teta_retour);
        end;
        goto menu;
CHOIX(7):
        DO;
        put list('Valeur : ');
        get list(alpha_initial);
        end;
        goto menu;
choix(8):
        do;
        put list ('Valeur : ');
        get list (alpha_final);
        end;
        goto menu;
CHOIX(9):
        DO;
        put list('Valeur : ');
        get list (ALPHA_DEL);
        end;
        goto menu;
CHOIX(10):
        DO;
        put list('Valeur : ');
        get list (increment_teta);
        call valmax2(increment_teta);
```

```
        end;
        goto menu;
CHOIX(11):
        DO;
        put list('Valeur : ');
        get list (NB_PASSES_INT);
        call valmax2(NB_PASSES_INT);
        end;
        goto menu;
CHOIX(12):
        DO;
        put list ('valeur : ');
        get list (del_val);
        END;
        GOTO MENU;
CHOIX(13):
        DO;
        call envoie();
        end;
        goto menu;
CHOIX(14):
        DO;
        OPEN FILE (acqpar) UPDATE DIRECT ENV(F(128));
        READ FILE(acqpar) INTO(para_source) KEY(0) ;
        READ FILE(acqpar) INTO(acq_parms) KEY(1) ;
        CLOSE FILE(acqpar);
        goto menu;
        end;
CHOIX(15):
        DO;
        OPEN FILE (acqpar) UPDATE DIRECT ENV(F(128));
        WRITE FILE(acqpar) FROM(para_source) KEYFROM(0) ;
        WRITE FILE(acqpar) FROM(acq_parms) KEYFROM(1) ;
        CLOSE FILE(acqpar);
        goto menu;
        end;
CHOIX(0):
        DO;
        PUT SKIP(3) LIST('***** MAUVAISE REPONSE *****');
        end;
        goto menu;
CHOIX(16):
        DO;
        OPEN FILE (acqpar) UPDATE DIRECT ENV(F(128));
        WRITE FILE(acqpar) FROM(para_source) KEYFROM(0) ;
        WRITE FILE(acqpar) FROM(acq_parms) KEYFROM(1) ;
        CLOSE FILE(acqpar);
        END;
        RETURN;
CHOIX(17):
        DO;
        put skip list('valeur : ');
        get list(trans);
        end;
        goto menu;
```

```
ENVOIE:
        PROCEDURE;

/* Initialise pour chaque mots de donnees les bits 14 et 15 */

        para7 = increment_teta * 16;
        para9 = NB_PASSES_INT* 16;

        valeur_binary = teta_initiale;
        valeur_bit = valeur_bit ! masque;
        para1 = valeur_binary;

        valeur_binary = phy_initiale;
        valeur_bit = valeur_bit ! masque;
        para2 = valeur_binary;

        valeur_binary = teta_final;
        valeur_bit = valeur_bit ! masque;
        para3 = valeur_binary;

        valeur_binary = phy_final;
        valeur_bit = valeur_bit ! masque;
        para4  = valeur_binary;

        valeur_binary = vitesse_teta_aller;
        valeur_bit = valeur_bit ! masque;
        para5 = valeur_binary;

        valeur_binary = vitesse_teta_retour;
        valeur_bit = valeur_bit ! masque;
        para6 = valeur_binary;

        valeur_binary = para7;
        valeur_bit = valeur_bit ! masque;
        para7 = valeur_binary;

        valeur_binary = vitesse_phy_aller;
        valeur_bit = valeur_bit ! masque;
        para8 = valeur_binary;

        valeur_binary = para9;
        valeur_bit = valeur_bit ! masque;
        para9 = valeur_binary;


/* Envoie les parametres a la camera */

        parametres->dim(1)=1;
        parametres->dim(2)=13;
        call dma(3,parametres);
END ENVOIE;


VALMAX1:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>4095) THEN
```

```
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 4095 ***');
        PUT SKIP LIST(' Donnez une autre ');
        GOTO CHOIX(INDEX('123456789PM',numero));
        END;
END VALMAX1;


VALMAX2:
        PROCEDURE (VALEUR);
DCL     VALEUR          BINARY;
        IF(VALEUR>255) THEN
        DO;
        PUT SKIP LIST('*** ERREUR : Valeur plus grande que 255 ***');
        PUT SKIP LIST(' Donnez une autre ');
        GOTO CHOIX(INDEX('123456789PM',numero));
        END;
END VALMAX2;


END DEFACQ;
```

```
PARAL:
        PROCEDURE ;

        %INCLUDE 'STRUC.LIB';

DCL
        DMA_PARMS                   POINTER,
        DONNEES                     POINTER;

DCL     COMP                        BINARY,
        NUMBLOC                     BINARY,
        D                           BINARY,
        DELAI                       DECIMAL,
        CANAL                       CHARACTER(1);

DCL     setdma          entry (POINTER),
        dma             entry (FIXED BIN(15),POINTER),
        convert         entry (pointer,pointer,decimal,binary);

DCL     ADDR_PTR                    POINTER,
        VALUE_IN_DECIMAL            FIXED DECIMAL(15,0),
        ADDR_BASE                   FIXED DECIMAL(15,0);

DCL     VPTR                        POINTER,
        VALUE_IN_BINARY             BINARY,
        VALUE_AS_POINTER            POINTER BASED (VPTR);

        VPTR=ADDR (VALUE_IN_BINARY);

        ALLOCATE WINDOW_STRUCTURE SET(DONNEES);

        CALL SETDMA(dma_parms);

        PUT SKIP LIST('Entree (0) ou sortie (1) de la carte 0    ');
        PUT SKIP LIST('Entree (2) ou sortie (3) de la carte 1    ');
        PUT SKIP(2) LIST ('Canal :      ');
        GET LIST (CANAL);

        PUT SKIP(2) LIST('Combien de blocs ? ');
        GET LIST (NUMBLOC);
        PUT SKIP LIST('Quel delai entre les transferts ? ');
        GET LIST (DELAI);
        PUT SKIP(2) LIST(' ');

        COMP=1;
        ADDR_BASE=131072;
        VALUE_IN_DECIMAL=ADDR_BASE;
        DONNEES->DIM(1)=1;
        DONNEES->DIM(2)=4096;

  DO WHILE(COMP <= NUMBLOC);
        CALL CONVERT(DONNEES,ADDR_PTR,VALUE_IN_DECIMAL,1);
        CALL DMA (CANAL,DONNEES);
        VALUE_IN_DECIMAL = VALUE_IN_DECIMAL + 8192;

        PUT SKIP(0) LIST('Transfert du ',COMP,'ieme bloc de 4 Kbytes');
        COMP=COMP+1;
```

```
/*      PETITE BOUCLE DE DELAI POUR ATTENDRE LE DMF-32 */
        D = 0 ;
        DO WHILE(D < DELAI) ;
                D = D + 1 ;
        END ;
END; /* do while */

END PARAL;
```

ANNEXE B

Sources des programmes de calibration sur VAX-11/750

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C ZCALI
C
C Ecrit par : Luc Cournoyer                              Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de recherche des parametres d'une fonction BSPLINE
C       representant la distorsion en "Z" du capteur 3D, a partir
C       d'un fichier de calibration en distance.
C
C       La fonction SPLINE cree a pour entree la position du pixel
C       dans l'image et la lecture de la barette CCD. La sortie de
C       la fonction est la valeur corrigee de la distance Z.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile,filepara
        integer znum,tnum,pnum
        integer*2 buffer,p(1024)
        real scl,xof,yof
        dimension xx(5000),yy(5000),w(5000),zz(5000),outmoy(1024),
     &  tx(100),ty(100),c(200),sum(1024),div(1024),outline(1024)
        common /option/no(30),fp0
        data ir,ip/10,6/
C
C Liste des formats
30      format(1a80)
40      format('0','smoothing factor s=',f10.0,5x,'error flag ier=',i2)
50      format('0','sum of squared residuals :',e15.6)
60      format(' La distance # ',I2,' a une valeur "Z" de ',F6.2,' unites')
280     format ('0','Fin de la correction de l''image ',3a1)
C
C Nombre de bits pour "p" a la sortie de la camera 3D (RETICON)
        nbits=13
C
C  Lecture du fichier de calibration et moyenne des lignes correspondantes
C  a chaque distance 'Z'.
C
        type*,' '
        type*,'                                   ZCALI '
        type*,' '
        type*,'                              Calibration "Z"'
        type*,' '
        type*,'Nom du fichier de calibration ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        type*,'Nombre de colonnes ? '
```

```
      read(5,*)tnum
      type*,'Nombre de lignes par position "Z" ? '
      read(5,*)pnum
      type*,'Nombre de positions "Z" differentes ? '
      read(5,*)znum
      type*,'Distance en Unite entre chaque position "Z" ?'
      read(5,*)deltaz
      type*,'Nom du fichier de parametres pour la calibration en "Z" ?'
      read(5,30) filepara
C
      type*,' '
      type*,'Effectue la moyenne des lignes pour chaque distance "Z" '
      type*,' '
      open(unit=3,name='ZMOY.DAT',type='new',access='sequential',
     & form='unformatted')
      kx = 0
      norg = 0
      xof = 0.0
      yof = 0.0
      grdnull = -99990.
      write(3) tnum,znum,sc1,kx,norg,xof,yof,grdnull,
     & xinc,yinc,(i1,j=11,126)
      m=0.0
      do i=1,znum
              v=0.0
              do k=1,tnum
                      sum(k)=0.0
                      div(k)=0.0
              enddo
              do j=1,pnum
                      read(1) (p(k),k=1,tnum)
                      do k=1,tnum
                              call mvbits(p(k),14,1,ibuf,0)
                              if (ibuf.eq.1) then
                                      call mvbits(p(k),0,13,buffer,0)
                                      if (buffer.lt.8191) then
                                              sum(k)=sum(k)+buffer
                                              div(k)=div(k)+1.0
                                              v=v+1.0
                                      endif
                              endif
                              if (j.eq.pnum) then
                                      if (div(k).eq.0) then
                                              outmoy(k)=0.0
                                      else
                                              m=m+1.0
                                              xx(m)=k
                                              yy(m)=sum(k)/div(k)
                                              zz(m)=(i+3.0)*deltaz
                                              outmoy(k)=yy(m)
                                      endif
                              endif
                      enddo
              enddo
```

```
                write (3) (outmoy(k),k=1,tnum)
                write(6,60) i,zz(m)
        enddo
        close (unit=1,status='keep')
C
C Le facteur d'echelle est utile seulement pour avoir
C un affichage sur plus de bits. Les donnees sont
C toujours en reel et en nombre d'unite(s).
        scl=32767.0/((znum+6.0)*deltaz)
        type*,' '
        type*,'Le facteur d''echelle est de : ',scl
        type*,' '
C
        type*,' '
        type*,'Le nombre total de points valides utilises'
        type*,'pour evaluer la fonction SPLINE est de : ',m
C
C Initialisation du tableau de 'WEIGHT'
        do k=1,m
                w(k) = 1.0
        enddo
C
C Determination du degre de l'aproximation SPLINE
        kx = 3
        ky = 3
C
C Determination des frontieres de l'aproximation SPLINE
        xb = 1.0
        xe = 1.0*tnum
        yb = 0.0
        ye = 2.0**(nbits)
C
C Determination du facteur d'amortissement
        type*,' '
        type*,'Calcul du facteur d''amortissement :'
        type*,'         0) a partir des donnees de calibration'
        type*,'         1) directement'
        read*, calcul
        if (calcul.eq.0) then
                call zfact(infile,znum,pnum,tnum,ecart)
                s = m*ecart
                type*,' '
                type*,'Le facteur d''amortissement est :',s
        else
                type*,'Quel est le facteur d''amortissement ? '
                read*,s
        endif
C
C Initialisation
        iopt = 0
C
        type*,'Determination de l''approximation SPLINE'
        call surfac1(xx,yy,zz,w,m,xb,xe,yb,ye,kx,ky,s,
     &  nx,tx,ny,ty,c,fp,iopt,ier)
```

```
C
      write(ip,40) s,ier
      write(ip,50) fp
C
      open(unit=2,name=filepara//'.PAR',type='new',access='sequential',
     & form='unformatted')
      write (2) nx,ny,kx,ky,xb,xe,yb,ye,sc1
      write (2) (tx(i),i=1,nx)
      write (2) (ty(i),i=1,ny)
      write (2) (c(i),i=1,(nx-kx-1)*(ny-ky-1))
      type*,'Nombre de noeuds (nx,ny) : ',nx,ny
C
C Fermeture des fichiers
      close (unit=2,status='save')
      close (unit=3,status='save')
      write(*,280) char(7),char(7),char(7)
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C XCALI1
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de recherche des parametres d'une fonction BSPLINE
C       representant la distorsion en "X" du capteur 3D, a partir
C       d'un fichier de calibration en intensite.
C
C       La fonction SPLINE intermediaire cree a pour entree la position
C       des franges en fraction de pixel d'image et 'a position du plan
C       de calibration en distance Z. La sortie de la fonction est la
C       position des franges en unites.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      character*80 infile,outfile
      integer maxcount(50),znum,tnum,pnum,columns,lines,oper
      integer*2 buffer,in(1024)
      dimension xp(50,50),zti(50,1024),
     & dep(1024),res1(1024),res2(1024),outligne(4,1024)
C
      dimension xx(5000),yy(5000),w(5000),zz(5000),
     & tx(100),ty(100),c(200),sum(1024)
      common /option/no(30),fp0
      data ir,ip/10,6/
C
C Liste des formats
30    format(1a80)
40    format('0','smoothing factor s=',f10.0,5x,'error flag ier=',i2)
50    format('0','sum of squared residuals :',e15.6)
60    format(' La distance # ',I2,' a une valeur Z de ',F6.2,' unites')
70    format(' Pour la distance # ',I2,
     & ' le nombre de maximums detectes est de ',F6.2)
80    format(' La frange # ',I2,' est a ',F6.2,
     & ' pixels du debut soit ',F6.2,' unites')
C
C  Lecture du fichier de calibration et moyenne des lignes correspondantes
C  a chaque distance 'Z'.
C
C
      type*,' '
      type*,'                                   XCALI '
      type*,' '
      type*,'                               Calibration "X"'
      type*,' '
      type*,'Nom du fichier de calibration ?'
      read(5,30) infile
      open(unit=1,name=infile,type='old',access='sequential',
     & form='unformatted')
```

```
      type*,'Nombre de colonnes ? '
      read(5,*)tnum
      type*,'Nombre de lignes par position "Z" ? '
      read(5,*)pnum
      type*,'Nombre de positions "Z" differentes ? '
      read(5,*)znum
      type*,'Distance en Unite entre chaque position "Z" ?'
      read(5,*)deltaz
      type*,'Distance en Unite entre chaque frange en "X" ?'
      read(5,*)delta
      type*,'Distance en Unite entre les deux franges centrales ?'
      read(5,*)deltax
      type*,'Details pour la verification (1=oui) ?'
      read*, verif
      type*,'Donnees a complementer (1=oui) ? '
      read*, inverse
C
C  Effectue la moyenne des lignes de donnees
      type*,' '
      type*,'Effectue la moyenne des lignes pour chaque distance "Z" '
      type*,' '
      do i=1,znum
            do k=1,tnum
                  sum(k)=0.0
            enddo
            do j=1,pnum
                  read(1) (in(k),k=1,tnum)
                  do k=1,tnum
                        call mvbits(in(k),1,8,buffer,0)
                        if (inverse.eq.1) buffer=256-buffer
                        sum(k)=sum(k)+buffer
                        if (j.eq.pnum) then
                              zti(i,k)=sum(k)/pnum
                        endif
                  enddo
            enddo
            write(6,60) i,(i+3.0)*deltaz
      enddo
C
      close (unit=1,status='keep')
C
C Fichier pour la verification de la detection des maximums
      open(unit=3,name='DERIV.DAT',type='new',access='sequential',
     &  form='unformatted')
      columns=tnum
      lines=znum*4.0
      kx = 0
      norg = 0
      xof = 0.0
      yof = 0.0
      grdnull = -99990.
      write(3) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(i1,j=11,126)
C
C Valeur du seuil de detection
200   type*,' '
      type*,'Valeur du seuil ? '
```

```
      read(5,*) th
C
C  Largeur de l'operateur (court)
      oper=2
C
C Pour compter le nombre maximum de franges detectes sur la cible
      maxfrange=0
C
C  Recherche des maximums d'intensite en utilisant l'operateur
C  [ 1 1 0 -1 -1 ] et application d'un seuil sur la derivee seconde
C  pour valider les passages par zero de la derivee premiere.
C
      do i=1,znum
      type*,' '
      count=0
C
C Calcul de la premiere derivee
      do j=1,tnum
            dep(j)=zti(i,j)
      enddo
      call deriv(dep,res1,tnum,oper)
C
C Calcul de la seconde derivee
C
      call deriv(res1,res2,tnum,oper)
C
C Determination du maximum et ecriture dans le fichier d'affichage
C
      min1=0
      min2=0
      do j=1,tnum
            if (res1(j).lt.min1) min1=res1(j)
            if (res2(j).lt.min2) min2=res2(j)
      enddo
      do j=1,tnum
            outligne(1,j)=zti(i,j)
            outligne(2,j)=res1(j)-min1
            outligne(3,j)=res2(j)-min2
            outligne(4,j)=th-min2
      enddo
      do l=1,4
            write(3) (outligne(l,j),j=1,tnum)
      enddo
C
C Trouve les transitions valides de la premiere derivee
C
      do j=1,tnum-1
      if(res1(j).ge.0.and.res1(j+1).lt.0.and.res2(j).le.th)then
            jp=j
            count=count+1
C
C Calcule la position exacte des maximums avec l'interpolation lineaire
C
            xp(i,count)=jp-res1(jp)/(res1(jp+1)-res1(jp))
            if (verif.eq.1) then
                  if (count.gt.1) then
                        write(6,*)count,xp(i,count),xp(i,count)-xp(i,count-1)
```

```
                        else
                                write(6,*)count,xp(i,count)
                        endif
                endif
        endif
        enddo
        if (count.gt.maxfrange) maxfrange=count
        maxcount(i)=count
        write(6,70) i,count
        enddo
C
C Petite boucle pour trouver le meilleur seuil pour la detection des franges
        type*,' '
        type*,' '
        type*,'La detection est-elle correcte (0=non,-1=stop,1=oui) ?'
        read(5,*) loop
        if (loop.eq.0) goto 200
        if (loop.eq.-1) goto 999
C
C Effectue la correspondance des centres des franges
C avec leur position en "X"
        m=0.0
        xnum=maxfrange+4
        do i=1,znum
220             format (' Position en Z # ',I2)
                write(6,220) i
                type*,' '
                count=maxcount(i)
                delmax=0.0
C
C Trouve la position de l'intervalle maximum
                do j=1,count-1
                        if((xp(i,j+1)-xp(i,j)).gt.delmax) then
                                delmax=xp(i,j+1)-xp(i,j)
                                posmax=j
                        endif
                enddo
C
C Fait la correspondance avec la valeur reelle de "X" et ecrit
C les parametres d'entree de la fonction d'interpolation SPLINE
                xr=(xnum/2 - posmax)*delta
                do j=1,count
                        if (j.eq.posmax+1) then
                                xr=xr+deltax
                        else
                                xr=xr+delta
                        endif
                        m=m+1.0
                        xx(m)=xp(i,j)
                        yy(m)=(i+3.0)*deltaz
                        zz(m)=xr
                        if (verif.eq.1) write(6,80) j,xx(m),zz(m)
                enddo
        enddo
C
C Determination du facteur d'echelle pour l'affichage de la table
        scl=32767.0/((xnum)*delta)
```

```
        type*,' '
      - type*,'Le facteur d''echelle X est de : ',scl
      -  type*,' '
C
        type*,' '
        type*,'Le nombre total de points valides utilises'
        type*,'pour evaluer la fonction SPLINE est de : ',m
C
C Initialisation du tableau de 'WEIGHT'
        do k=1,m
                w(k) = 1.0
        enddo
C
C Determination du degre de l'aproximation SPLINE
        kx = 3
        ky = 3
C
C Determination des frontieres de l'aproximation SPLINE
        xb = 0.0
        xe = 1.0*tnum
        yb = 0.0
        ye = (znum+6.0)*deltaz
C
C Determination du facteur d'amortissement
        type*,' '
        type*,' '
        type*,'Calcul du facteur d''amortissement :'
        type*,'          0) a partir des donnees de calibration'
        type*,'          1) directement'
        read*, calcul
        if (calcul.eq.0) then
                call xfact(infile,znum,pnum,tnum,th,ecart)
                s = m*ecart
                type*,' '
                type*,'Le facteur d''amortissement est :',s
        else
                type*,'Quel est le facteur d''amortissement ? '
                read*,s
        endif
C
C Initialisation
        iopt = 0
C
        type*,'Determination de l''approximation SPLINE'
        call surfac1(xx,yy,zz,w,m,xb,xe,yb,ye,kx,ky,s,
     &  nx,tx,ny,ty,c,fp,iopt,ier)
C
        write(ip,40) s,ier
        write(ip,50) fp
C
        open(unit=2,name='TEMP.DAT',type='new',access='sequential',
     &  form='unformatted')
        write (2) tnum,znum,deltaz
        write (2) nx,ny,kx,ky,xb,xe,yb,ye,scl
        write (2) (tx(i),i=1,nx)
        write (2) (ty(i),i=1,ny)
        write (2) (c(i),i=1,(nx-kx-1)*(ny-ky-1))
```

```
      type*,'Nombre de noeuds (nx,ny) : ',nx,ny
C
C Fermeture des fichiers
999   close (unit=2,status='save')
      close (unit=3,status='save')
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C XCAL12
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de recherche des parametres d'une fonction BSPLINE
C       representant la distorsion en "X" du capteur 3D, a partir
C       de la fonction SPLINE intermediaire et d'un fichier de
C       calibration en distance.
C
C       La fonction SPLINE cree a pour entree la position du pixel
C       dans l'image et la lecture de l'image de calibration en
C       distance Z. La sortie de la fonction est la valeur corrigee
C       de la distance Z, donnee par la fonction SLPINE intermediaire.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,filepara
        integer znum,tnum,pnum,zznum,ttnum
        dimension xx(5000),yy(5000),w(5000),zz(5000),zp(1024),
     &  tx(100),ty(100),c(200),sum(1024),outline(1024),intligne(4,1024)
        common /option/no(30),fp0
        data ir,ip/10,6/
C
C Liste des formats
30      format(1a80)
40      format('0','smoothing factor s=',f10.0,5x,'error flag ier=',i2)
50      format('0','sum of squared residuals :',e15.6)
280     format ('0','Fin de la correction de l''image ',3a1)
C
C Nombre de bits pour "p" a la sortie de la camera 3D
        nbits=13
C
C Lecture des parametres de la fonction SPLINE
        open(unit=1,name='TEMP.DAT',type='old',access='sequential',
     &  form='unformatted')
        read (1) tnum,znum,deltaz
        read (1) nx,ny,kx,ky,xb,xe,yb,ye,sc1
        read (1) (tx(i),i=1,nx)
        read (1) (ty(i),i=1,ny)
        read (1) (c(i),i=1,(nx-kx-1)*(ny-ky-1))
        close (unit=1,status='delete')
C
C Fichier de calibration en "Z"
        open(unit=2,name='ZMOY.DAT',type='old',access='sequential',
     &  form='unformatted')
        read (2) ttnum,zznum
```

```
      m=0.0
      do i=1,znum
            read (2) (zp(k),k=1,ttnum)
            do j=1,tnum
                  m=m+1.0
                  x=j
                  y=(i+1.0)*deltaz
                  xx(m)=x
                  yy(m)=zp(j)
                  zz(m)=bisp(tx,nx,ty,ny,kx,ky,c,x,y)
            enddo
      enddo
C
      type*,' '
      type*,'Le nombre total de points valides utilises'
      type*,'pour evaluer la fonction SPLINE est de : ',m
      type*,' '
C
C Initialisation du tableau de 'WEIGHT'
      do k=1,m
            w(k) = 1.0
      enddo
C
C Determination du degre de l'aproximation SPLINE
      kx = 3
      ky = 3
C
C Determination des frontieres de l'aproximation SPLINE
      xb = 0.0
      xe = 1.0*tnum
      yb = 0.0
      ye = 2.0**(nbits)
C
C Determination du facteur d'amortissement
      type*,' '
      type*,'Quelle est la valeur du bruit sur les donnees ? '
      read*,bruit
      s = m*bruit**2
      type*,' '
      type*,'Le facteur d''amortissement est :',s
      type*,'Nom du fichier de parametres pour la calibration en "X" ?'
      read(5,30) filepara
C
C Initialisation
      iopt = 0
C
      type*,'Determination de l''approximation SPLINE'
      call surfac1(xx,yy,zz,w,m,xb,xe,yb,ye,kx,ky,s,
     & nx,tx,ny,ty,c,fp,iopt,ier)
C
      write(ip,40) s,ier
      write(ip,50) fp
      open(unit=3,name=filepara//'.par',type='new',access='sequential',
     & form='unformatted')
```

```
      write (3) nx,ny,kx,ky,xb,xe,yb,ye,scl
      write (3) (tx(i),i=1,nx)
      write (3) (ty(i),i=1,ny)
      write (3) (c(i),i=1,(nx-kx-1)*(ny-ky-1))
      type*,'Nombre de noeuds (nx,ny) : ',nx,ny
C
C Fermeture des fichiers
      close (unit=2,status='save')
      close (unit=3,status='save')
      write(*,280) char(7),char(7),char(7)
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C GENTAB
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de generation d'un fichier representant une table de
C       calibration pour fins de verification. Cette table est interpolee
C       a l'aide de la sous-routine BISP et d'un fichier intermediaire
C       qui represente la distorsion en X ou en Z de la camera, qui
C       ont ete produits lors de la calibration.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*2 buffer,p(1024)
        integer lines, columns, kx, norg
        real*4 scl, xof, yof, grdnull, z2(512,512)
        dimension xxx(512,512),zzz(512,512),zout(1024)
        dimension tx1(100),ty1(100),c1(200),tx2(100),ty2(100),c2(200)
        common /option/no(30),fp0
        data ir,ip/10,6/
C
C Liste des formats
30      format (1a80)
210     format ('+','Generation de la ligne #',i3)
280     format ('0','Fin de la correction de l''image ',3a1)
C
C Lecture des parametres de la fonction SPLINE
        type*,' '
        type*,'Quel fichier contient les parametres de correction "Z" ?'
        read(*,30) infile
        open(unit=1,name=infile//'.par',type='old',access='sequential',
     &  form='unformatted')
        read (1) nx1,ny1,kx1,ky1,xb1,xe1,yb1,ye1,scl
        read (1) (tx1(i),i=1,nx1)
        read (1) (ty1(i),i=1,ny1)
        read (1) (c1(i),i=1,(nx1-kx1-1)*(ny1-ky1-1))
        close (unit=1,status='keep')
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
        type*,'Quel fichier contient les parametres de correction "X" ?'
        read(*,30) infile
        open(unit=2,name=infile//'.par',type='old',access='sequential',
     &  form='unformatted')
        read (2) nx2,ny2,kx2,ky2,xb2,xe2,yb2,ye2
        read (2) (tx2(i),i=1,nx2)
        read (2) (ty2(i),i=1,ny2)
        read (2) (c2(i),i=1,(nx2-kx2-1)*(ny2-ky2-1))
        close (unit=2,status='keep')
```

```
      open(unit=2,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
      type*,'Combien de lignes d''affichage '
      read*, lines
      if (ye1.ne.ye2 .or. xe1.ne.xe2) then
            type*,'*** ERREUR *** Limites des fonctions spline differentes'
            stop
      endif
      yinc=ye2/lines
      columns=INT(xe2)
C
C Correction des donnees a partir des parametres et determination
C des maximums et minimums en "X" pour l'interpolation finale
C
      y = 0.0
      do i=1,lines
            write (*,210) i
            y = y + yinc
            do j=1,columns
                  x = j
                  zzz(i,j) = bisp(tx1,nx1,ty1,ny1,kx1,ky1,c1,x,y)
                  xxx(i,j) = bisp(tx2,nx2,ty2,ny2,kx2,ky2,c2,x,y)
            enddo
      enddo
C
C Ecriture des facteurs dans le fichier de l'image corrigee
      kx = 0
      norg = 0
      xof = 0.0
      yof = 0.0
      grdnull = -99990.
      write(1) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  (i1,j=9,126)
      write(2) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  (i1,j=9,126)
C
C Ecriture des donnees interpolees
      do i=1,lines
            write (1) (zzz(i,j)*scl,j=1,columns)
            write (2) (xxx(i,j)*scl,j=1,columns)
      enddo
C
C Fermeture des fichiers
      close (unit=1,status='save')
      close (unit=2,status='save')
C
      write(*,280) char(7),char(7),char(7)
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C ZFACT
C                                                          Fevrier 1986
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme de calcul de l'erreur quadratique moyen a partir
C       du fichier de calibration en "Z". Retourne une evaluation du
C       facteur d'amortissement au programme de calibration ZCALI.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      subroutine zfact(infile,znum,pnum,tnum,ecart)
C
      character*80 infile
      integer*2 buffer,p(64,512)
      integer znum,tnum,pnum
      real vectemp(64)
C
30    format(1a80)
50    format ('+','Calcul pour la position #',I3)
C
      open(unit=1,name=infile,type='old',access='sequential',
     & form='unformatted')
C
      ecartsum=0.0
      do i=1,znum
            write (6,50) i
            do j=1,pnum
                  read(1) (p(j,k),k=1,tnum)
            enddo
            do k=1,tnum
                  m=0
                  do j=1,pnum
                        call mvbits(p(j,k),14,1,ibuf,0)
                        if (ibuf.eq.1) then
                              call mvbits(p(j,k),0,13,buffer,0)
                              m=m+1
                              vectemp(m)=buffer
                        endif
                  enddo
                  if (m.ne.0) then
                        call ecaqua(vectemp,m,ecart)
                        ecartsum = ecartsum+ecart
                  endif
            enddo
      enddo
      ecart = ecartsum/(znum*tnum)
100   format('0','L''ecart quadratique moyen est de ',F10.4)
      write (6,100) ecart
C
```

```
      close (unit=1,status='keep')
C
      return
      end
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     Sous-routine pour le calcul de l'ecart quadratique moyen
C
      subroutine ecaqua(vector,m,ecart)
C
      integer m
      real vector(64)
C
C calcul de la somme
      valmoy = 0.0
      do i=1,m
            valmoy = valmoy + vector(i)
      enddo
      valmoy = valmoy/m
C
C calcul de l'ecart quadratique
      do i=1,m
            sum = (valmoy-vector(i))**2
      enddo
      ecart = sum/(m-1)
C
      return
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C XFACT
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme de calcul de l'erreur quadratique moyen a partir
C       du fichier de calibration en "X". Retourne une evaluation du
C       facteur d'amortissement au programme de calibration XCALI1.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        subroutine xfact(infile,znum,pnum,tnum,th,ecart)
C
        character*80 infile
        integer znum,tnum,pnum,oper,count
        integer*2 buffer,in(512)
        dimension xp(50,50),dep(512),res1(512),res2(512),vectemp(64)
C
50      format ('+','Position # ',I3,', nombre de franges = ',I3)
C
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
C  Largeur de l'operateur
        oper=2
C
        ecartotal = 0.0
        do i=1,znum
                ecartsum = 0.0
                do j=1,pnum
                        read(1) (in(k),k=1,tnum)
                        do k=1,tnum
                                call mvbits(in(k),1,8,buffer,0)
                                dep(k)=buffer
                        enddo
                        call deriv(dep,res1,tnum,oper)
                        call deriv(res1,res2,tnum,oper)
                        count=0
                        do k=1,tnum-1
                       if(res1(k).ge.0.and.res1(k+1).lt.0.and.res2(k).le.th)then
                                jp=k
                                count=count+1
                                xp(j,count)=jp-res1(jp)/(res1(jp+1)-res1(jp))
                        endif
                        enddo
                enddo
                do k=1,count
                        do j=1,pnum
                                vectemp(j)=xp(j,k)
                        enddo
```

```
                        call ecaqua(vectemp,pnum,ecart)
                        ecartsum = ecartsum + ecart
                enddo
                ecartotal = ecartotal + ecartsum/count
                write (6,50) i,count
        enddo
        ecart = ecartotal/znum
100     format('0','L''ecart quadratique moyen est de ',F10.4)
        write (6,100) ecart
C
        close (unit=1,status='keep')
C
        return
        end
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C       Sous-routine pour le calcul de l'ecart quadratique moyen
C
        subroutine ecaqua(vector,m,ecart)
C
        integer m
        real vector(64)
C
C calcul de la somme
        valmoy = 0.0
        do i=1,m
                valmoy = valmoy + vector(i)
        enddo
        valmoy = valmoy/m
C
C calcul de l'ecart quadratique
        do i=1,m
                sum = (valmoy-vector(i))**2
        enddo
        ecart = sum/(m-1)
C
        return
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C DERIV
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme de derivation d'un tableau DEP de N elements
C       avec un operateur de dimension 2xOP. Le tableau de retour
C       est RES.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        subroutine deriv (dep,res,n,op)
C
        integer n,op
        real dep(1024),int(1024),res(1024)
C
C Fill with zero's
        do i=1,n+2*op
                int(i)=0.0
                res(i)=0.0
        enddo
C
C Shift the array by OP pixels
        do i=1,n
                int(i+op)=dep(i)
        enddo
C
C Summation of the terms
        do i=op+1,n+op
                do j=1,op
                        res(i)=res(i)+int(i+j)-int(i-j)
                enddo
        enddo
C
C Shift back as the original
        do i=1,n
                res(i)=res(i+op)
        enddo
C
        return
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C YCALI
C
C Ecrit par : Luc Cournoyer                                   Mai 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme d'evaluation de la valeur de H qui sera utilise pour
C       la correction des images en Y. Le fichier de calibration qui est
C       requis pour cette evaluation est une image avec balayage en Y
C       d'un plan de reference pris a une position en Z qui soit tres
C       bien connue.
C
C       L'evaluation de H est faite a partir d'une moyenne ponderee
C       statistique des valeurs de H calcules pour chaque pixels a
C       l'interieur de la plage consideree.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer row,col,ier,dpnum
        real phistart,delphi,phimarge,nbpoint,nbpointotal
        real*8 cosphi(512)
        dimension h(100000),histo(100000),zin(512)
30      format (1a80)
280     format (3a1)
C
C Lecture des parametres de depart
C
        type*,' '
        type*,'         Evaluation du parametre H pour la correction en "Y"'
        type*,' '
        type*,'Quel est le fichier a utiliser pour l''evaluation de H ?'
        read(5,30) infile
        open(unit=1,name=infile//'.xin',type='old',access='sequential',
     &  form='unformatted')
        read (1) col,row,scl,kx,norg,xof,yof,grdnull,xinc
        type*,'Quelle distance "Z" cette image represente en unite(s) ?'
        read(5,*)zed
        type*,'Nom du fichier pour garder ces valeurs :'
        read(5,30)outfile
        open(unit=2,name=outfile//'.par',type='new',access='sequential',
     &  form='unformatted')
        type*,'Largeur de la marge de securite autour de l''image (en pixels)?'
        read*,marge
        type*,'Marge angulaire de securite autour de chaque cote'
        type*,'de la position Phi0 ,en degre(s) ?'
        read*,phimarge
99      type*,'Valeur de depart pour Phy ?'
        read(5,*)phistart
        type*,'Valeur de l''increment angulaire entre deux lignes ?'
        read(5,*)delphi
```

```
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
        phistart=phistart*4.0*atan(1.0)/180.0
        delphi=delphi*4.0*atan(1.0)/180.0
        phimarge=phimarge*4.0*atan(1.0)/180.0
        do i=1,row
                if (phistart+(i-1)*delphi.ge.-phimarge.and.
     &          phistart+(i-1)*delphi.le.phimarge) then
                        cosphi(i) = 1.0
                else
                        cosphi(i) = cos(phistart+(i-1)*delphi)
                endif
        enddo
C
        valnum = 0.0
        rewind (1)
        read (1) col,row,scl,kx,norg,xof,yof,grdnull,xinc
        do i=1,row
                read(1)(zin(j),j=1,col-marge)
                if (i.ge.marge) then
                        do j=marge,col-marge
                         if (cosphi(i) .ne. 1.0) then
                                valnum = valnum + 1.0
                                h(valnum) = (zed - zin(j)*cosphi(i)/scl)/
     &                                          (1.0 - cosphi(i))
                         endif
                        enddo
                endif
        enddo
C
C Affiche les valeurs pour l'histogramme
50      type*,'Valeurs minimum et maximum de l''histogramme ?'
        read*,valmin,valmax
        type*,'Valeur de l''increment ?'
        read*,valinc
        valhisto = 0
        valhistom = 0
        valhistoc = 0
        hmoy = 0.0
        hmoyp = 0.0
        nbpointotal = 0
        do valint=valmin,valmax-valinc,valinc
                nbpoint = 0
                do i=1,valnum
                        if (h(i).ge.valint.and.h(i).lt.valint+valinc) then
                                nbpoint = nbpoint+1
                                valhisto = valhisto+1
                                histo(valhisto) = h(i)
                        endif
                enddo
                type*,'Entre',valint,' et',valint+valinc
     &                                  ,'le nombre de points est',nbpoint
                valhistoc = valhistoc + nbpoint**2
                do i=valhistom+1,valhisto
```

```
                hmoy = hmoy + histo(i)
                hmoyp = hmoyp + histo(i)*nbpoint
            enddo
            valhistom = valhisto
        enddo
C
        type*,' '
        type*,'Pour cet intervalle :'
        type*,'       - le nombre de point total :',valhisto
        type*,'       - valeur moyenne de H :',hmoy/valhisto
        type*,'       - valeur moyenne ponderee de H :',hmoyp/valhistoc
        type*,' '
        type*,'Un autre histogramme (-1=changement,1=oui) ?'
        read*,autre
        if (autre.eq.-1) goto 99
        if (autre.eq.1) goto 50
C
        write(2) hmoyp/valhistoc,phistart,delphi
        close(unit=1,status='keep')
        close(unit=2,status='save')
C
        end
```

ANNEXE C

Sources des programmes de correction sur VAX-11/750

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C XCOR
C
C Ecrit par : Luc Cournoyer                                 Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de correction des donnees 3D d'un fichier a l'aide
C       des parametres generes par les programmes de calibration. Les
C       nouvelles donnees sont interpolees a l'aide du sous-programme
C       BISP et des fichier contenant les parametres.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*2 buffer,p(1024)
        integer lines, columns, kx, norg
        real*4 scl, xof, yof, grdnull, z2(512,512)
        dimension xxx(512,512),zzz(512,512),zout(1024)
        dimension tx1(100),ty1(100),c1(200),tx2(100),ty2(100),c2(200)
        common /option/no(30),fp0
        data ir,ip/10,6/
C
C Liste des formats
30      format (1a80)
210     format ('+','Correction de la ligne #',i3)
280     format ('0','Fin de la correction de l''image ',3a1)
C
C Nombre de bits pour "p" a la sortie de la camera 3D
        nbits=13
C
        type*,' '
        type*,'                    Correction des donnees 3D'
        type*,' '
C
        type*,'Quel fichier est a corriger ?'
        read(*,30) infile
        open(unit=3,name=infile,type='old',access='sequential',
     &  form='unformatted')
        type*,'Quel sont les dimensions (x,y) ?'
        read*,columns,lines
        open(unit=4,name=infile//'.XCO',type='new',access='sequential',
     &  form='unformatted')
C
C Lecture des parametres de la fonction SPLINE
        type*,'Quel fichier contient les parametres de correction "Z" ?'
        read(*,30) infile
        open(unit=1,name=infile//'.par',type='old',access='sequential',
     &  form='unformatted')
        read (1) nx1,ny1,kx1,ky1,xb1,xe1,yb1,ye1,scl
        read (1) (tx1(i),i=1,nx1)
        read (1) (ty1(i),i=1,ny1)
```

```
      read (1) (c1(i),i=1,(nx1-kx1-1)*(ny1-ky1-1))
      close (unit=1,status='keep')
C
      type*,'Quel fichier contient les parametres de correction "X" ?'
      read(*,30) infile
      open(unit=2,name=infile//'.par',type='old',access='sequential',
     & form='unformatted')
      read (2) nx2,ny2,kx2,ky2,xb2,xe2,yb2,ye2
      read (2) (tx2(i),i=1,nx2)
      read (2) (ty2(i),i=1,ny2)
      read (2) (c2(i),i=1,(nx2-kx2-1)*(ny2-ky2-1))
      close (unit=2,status='keep')
C
C Correction des donnees a partir des parametres et determination
C des maximums et minimums en "X" pour l'interpolation finale
C
      faux = 0.0
      do i=1,lines
            write (*,210) i
            read (3) (p(k),k=1,columns)
            do j=1,columns
                  call mvbits(p(j),14,1,ibuf,0)
                  if (ibuf.eq.1) then
                        call mvbits(p(j),0,13,buffer,0)
                        x = j
                        y = buffer
                        zzz(i,j) = bisp(tx1,nx1,ty1,ny1,kx1,ky1,c1,x,y)
                        xxx(i,j) = bisp(tx2,nx2,ty2,ny2,kx2,ky2,c2,x,y)
                        if (i.eq.1.and.j.eq.1) then
                              xmin=xxx(i,j)
                              xmax=xxx(i,j)
                        endif
                        if (xxx(i,j).lt.xmin) xmin=xxx(i,j)
                        if (xxx(i,j).gt.xmax) xmax=xxx(i,j)
                  else
                        xxx(i,j) = 0.0
                        zzz(i,j) = 0.0
                        faux = faux + 1
                  endif
            enddo
      enddo
C
C Determination des limites de l'aproximation SPLINE
      xinc = (xmax-xmin)/columns
      type*,'xmin = ',xmin,'xmax = ',xmax,'xinc = ',xinc
      type*,' '
      type*,'Le nombre de points non valides est de',faux
C
C Ecriture des facteurs dans le fichier de l'image corrigee
      kx = 0
      norg = 0
      xof = 0.0
      yof = 0.0
      grdnull = -99990.
      write(4) columns,lines,scl,kx,norg,xof,yof,grdnull,
     & xinc,yinc,xmin,xmax,(i1,j=13,126)
C
```

```
C Ecriture des donnees interpolees
      do i=1,lines
            k=1
            do j=1,columns
                  zout(k)=xxx(i,j)
                  zout(k+1)=zzz(i,j)*scl
                  k = k + 2
            enddo
            write (4) (zout(k),k=1,2*columns)
      enddo
C
C Fermeture des fichiers
      close (unit=3,status='keep')
      close (unit=4,status='save')
C
      write(*,280) char(7),char(7),char(7)
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C YCOR
C
C Ecrit par : Luc Cournoyer                                   Mars 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C
C       Programme de correction des donnees en fonction de leur
C       distance et de l'angle du balayage en Y. La correction est
C       faite a partir des parametres optimises par YCALI et d'une
C       fonction trigonometrique simple.
C
C       Le fichier d'entree est celui corrige en X et Z par le programme
C       XCOR. Le format du fichier de sortie est compatible avec l'editeur
C       de solide de la section d'infographie. Ce format consiste en un
C       registre d'information sur les dimensions et les facteurs d'echelle,
C       suivis de trois tableaux X, Y, et Z. Les dimensions de ces tableaux
C       sont les memes que celles de l'image originale et les coordonnees X,
C       Y et Z des points non valides sont remplaces par la valeur de zero.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      integer row,col
      real zin(1024),imax(512,512),imay(512,512),imaz(512,512)
      character*80 infile,parafile
      real phi(512),phicos(512),phisin(512),dely,ymin,ymax,y,phistart
C
C Liste des formats
30    format (1a80)
240   format ('+','Correction en Y pour la ',I3,' ieme ligne')
260   format ('0','Le facteur d''echelle en Z est de ',F10.4)
C
C Lecture du fichier a corriger pour l'affichage
C
      type*,' '
      type*,'                    Correction en "Y" '
      type*,' '
      type*,'Quel est le nom du fichier (X,Z) a corriger ?'
      read(5,30)infile
      open(unit=1,name=infile//'.XCO',type='old',access='sequential',
     & form='unformatted')
      read (1) col,row,scl,kx,norg,xof,yof,grdnull
     & ,xinc,yinc,xmin,xmax
      open(unit=2,name=infile//'.YCO',type='new',access='sequential',
     & form='unformatted')
      type*,'Parametres a partir d''un fichier (1=oui) ?'
      read*,para
      if(para.eq.1) then
            type*,'Quel est le nom du fichier de parametres optimises ?'
            read(*,30) parafile
            open(unit=3,name=parafile//'.par',type='old',access='sequential',
```

```
     &          form='unformatted')
                read(3) h,phistart,delphi
                close (unit=3,status='keep')
                type*,'Distance H = ',h
                type*,'Angle Phi de depart = ',phistart/(4.0*atan(1.0)/180.0)
                type*,'Increment angulaire = ',delphi/(4.0*atan(1.0)/180.0)
                type*,' '
        else
                type*,' '
                type*,'Valeur de la distance H en unites ?'
                read*,h
                type*,'Valeur de l''angle de depart ?'
                read*,phistart
                phistart = phistart*4.0*atan(1.0)/180.0
                type*,'Valeur de l''increment angulaire ?'
                read*,delphi
                delphi = delphi*4.0*atan(1.0)/180.0
                type*,' '
        endif
C
        type*,'Dimensions (x,y) : ',col,row
        write(6,260) scl
        type*,' '
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
        do i=1,row
                phicos(i) = cos(phistart + (i-1)*delphi)
                phisin(i) = sin(phistart + (i-1)*delphi)
        enddo
C
C  Calibre l'image dans la direction "Y"
C
        do i=1,row
                write (6,240) i
                read(1)(zin(j),j=1,2*col)
                do j=1,col
                        if (zin(2*j).ne.0) then
                                v = v + 1.0
                                imax(i,j) = zin(2*j-1)
                                imay(i,j) = (h-zin(2*j)/scl)*phisin(i)
                                imaz(i,j) = h - (h-zin(2*j)/scl)*phicos(i)
                        else
                                imax(i,j) = 0.0
                                imay(i,j) = 0.0
                                imaz(i,j) = 0.0
                        endif
                enddo
        enddo
C
        type*,' '
        type*,'Le nombre de points valides est de ',v,' sur ',col*row
        type*,'soit ',(v*100.0)/(col*row),' % des points'
        type*,' '
        type*,'Ecriture des donnees dans le fichier de sortie'
```

```
      write (2) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
      write (2) ((imax(i,j),j=1,col),i=1,row)
      write (2) ((imay(i,j),j=1,col),i=1,row)
      write (2) ((imaz(i,j),j=1,col),i=1,row)
      write(6,500) char(7),char(7),char(7)
      type*,'Fichiers DPICT X,Y et Z (1=oui) ?'
      read*,dpict
      if (dpict.eq.1) then
              open(unit=3,name=infile//'.CAX',type='new',
     &        access='sequential',form='unformatted')
              write (3) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
              open(unit=4,name=infile//'.CAY',type='new',
     &        access='sequential',form='unformatted')
              write (4) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
              open(unit=5,name=infile//'.CAZ',type='new',
     &        access='sequential',form='unformatted')
              write (5) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
              do i=1,row
              write (3) ((imax(i,j),j=1,col))
              write (4) ((imay(i,j),j=1,col))
              write (5) ((imaz(i,j),j=1,col))
              enddo
       close (unit=3,status='save')
       close (unit=4,status='save')
       close (unit=5,status='save')
       endif
C
       close (unit=1,status='save')
       close (unit=2,status='save')
C
       write(6,500) char(7),char(7),char(7)
500    format (3a1)
C
       end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C YCOR1
C
C Ecrit par : Luc Cournoyer                                  Aout 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C
C       Programme de correction des donnees en fonction de leur
C       distance et de l'angle du balayage en Y. La correction est
C       faite a partir des parametres optimises par YCALI et d'une
C       fonction trigonometrique simple. Le format du fichier de sortie
C       est DPICT BINAIRE XYZ ou ASCII en raison d'un triplet XYZ par
C       ligne.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        integer row,col
        real zin(1024),imaz(512),imay(512)
        character*80 infile
        real phi(512),phicos(512),phisin(512),dely,ymin,ymax,y,phistart
C
C Liste des formats
30      format (1a80)
240     format ('+','Correction en Y pour la ',I3,' ieme ligne')
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format ('0','La dimension des pixels en X est de ',
     &  F10.5,' unite(s)')
300     format(1X,3(F7.2))
C
C Lecture du fichier a corriger pour l'affichage
C
        type*,' '
        type*,'                    Correction en "Y" '
        type*,' '
        type*,'Quel est le nom du fichier (X,Z) a corriger ?'
        read(5,30)infile
        open(unit=1,name=infile//'.XCO',type='old',access='sequential',
     &  form='unformatted')
        read (1) col,row,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax
        type*,'Parametres a partir d''un fichier (1=oui) ?'
        read*,para
        if(para.eq.1) then
                type*,'Quel est le nom du fichier de parametres optimises ?'
                read(*,30) infile
                open(unit=3,name=infile//'.par',type='old',access='sequential',
     &          form='unformatted')
                read(3) h,phistart,delphi
                close (unit=3,status='keep')
                type*,'Distance H = ',h
                type*,'Angle Phi de depart = ',phistart/(4.0*atan(1.0)/180.0)
```

```
                type*,'Increment angulaire = ',delphi/(4.0*atan(1.0)/180.0)
                type*,' '
        else
                type*,' '
                type*,'Valeur de la distance H en unites ?'
                read*,h
                type*,'Valeur de l''angle de depart ?'
                read*,phistart
                phistart = phistart*4.0*atan(1.0)/180.0
                type*,'Valeur de l''increment angulaire ?'
                read*,delphi
                delphi = delphi*4.0*atan(1.0)/180.0
                type*,' '
        endif
C
        type*,'Dimensions (x,y) : ',col,row
        write(6,260) scl
        type*,' '
        type*,' '
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
        do i=1,row
                phicos(i) = cos(phistart + (i-1)*delphi)
                phisin(i) = sin(phistart + (i-1)*delphi)
        enddo
C
C  Calibre l'image dans la direction "Y"
C
        type*,'Ecriture binaire (1) ou ASCII (2) ?'
        read*,ecrit
        if (ecrit.eq.1) then
        open(unit=2,name=infile//'.CAB',type='new',access='sequential',
     &  form='unformatted')
        do i=1,row
                write (6,240) i
                read(1)(zin(j),j=1,2*col)
                do j=1,col
                        if (zin(2*j).ne.0) then
                                imay(j) = (h-zin(2*j)/scl)*phisin(i)
                                imaz(j) = h - (h-zin(2*j)/scl)*phicos(i)
                                write (2) zin(2*j-1),imay(j),imaz(j)
                        endif
                enddo
        enddo
        else
        open(unit=2,name=infile//'.CAA',type='new',form='formatted')
        do i=1,row
                write (6,240) i
                read(1)(zin(j),j=1,2*col)
                do j=1,col
                        if (zin(2*j).ne.0) then
                                imay(j) = (h-zin(2*j)/scl)*phisin(i)
                                imaz(j) = h - (h-zin(2*j)/scl)*phicos(i)
                                write (2,300) zin(2*j-1),imay(j),imaz(j)
```

```
                        endif
                enddo
        enddo
        endif
C
        close (unit=1,status='save')
        close (unit=2,status='save')
C
        write(6,500) char(7),char(7),char(7)
500     format (3a1)
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C XINT
C
C Ecrit par : Luc Cournoyer                            Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme d'interpolation lineaire dans l'axe X d'un fichier
C       deja corrige par XCOR. Les donnees sont interpolees sur une
C       grille reguliere pour permettre un affichage sur ecran graphique.
C       Les bornes de l'interpolation sont les mininums et maximums
C       rencontres lors de la correction.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer lines, columns, kx, norg
        real*4 scl, xof, yof, grdnull
        dimension vecin(1024,1124),zout(1024)
C
C Liste des formats
30      format (1a80)
250     format ('+','Interpolation de la ',I4,' ieme ligne')
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format (' La dimension des pixels en X est de ',F10.5,' unite(s)')
280     format (' L''affichage en X se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
300     format (3a1)
C
        type*,' '
        type*,'       Interpolation lineaire en "X" pour l''affichage'
        type*,' '
        type*,'Quel est le nom du fichier (X,Z) a interpoler ?'
        read(*,30) infile
        type*,' '
        open(unit=1,name=infile//'.XCO',type='old',access='sequential',
     &  form='unformatted')
        read (1) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,xmin,xmax
        open(unit=2,name=infile//'.XIN',type='new',access='sequential',
     &  form='unformatted')
C
C Determination des maximums et mimimums dans l'axe des "X"
        type*,' '
        type*,'Dimensions de l''image (x,y) : ',columns,lines
        write(*,260) scl
        write(*,270) xinc
        write(*,280) xmin,xmax
        write(2) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,xmin,xmax,(i1,j=13,126)
        type*,' '
        type*,' '
```

```
C
C Interpolation lineaire
      do i=1,lines
            write (*,250) i
            read(1) (vecin(i,k),k=1,2*columns)
C On trouve la premiere valeur valide de la ligne
            k = 1
            do while (vecin(i,k).eq.0)
                  k = k + 2
            enddo
            inival = k
C Depart de la boucle d'interpolation pour la ligne
            kd = inival
            x = xmin
            do j=1,columns
                  k = kd
                  if (x.lt.vecin(i,inival)) then
                        zout(j) = 0.0
                  else
                        do while (vecin(i,k).lt.x)
                              kd = k
                              k = k + 2
                              if (k.gt.2*columns) then
                                    zout(j) = 0.0
                                    goto 999
                              endif
                              do while (vecin(i,k).eq.0)
                                    k = k + 2
                                    if (k.gt.2*columns) then
                                          zout(j) = 0.0
                                          goto 999
                                    endif
                              enddo
                        enddo
                        if (k.eq.inival) then
                              zout(j) = vecin(i,k+1)
                        else
            slope = (vecin(i,k+1)-vecin(i,kd+1))/(vecin(i,k)-vecin(i,kd))
            zout(j) = (slope*x) + vecin(i,k+1) - (slope*vecin(i,k))
                        endif
                  endif
999               x = x + xinc
            enddo
            write (2) (zout(m),m=1,columns)
      enddo
1     format(3I12)
2     format(3f12.4)
C
C Fermeture des fichiers
      close (unit=1,status='keep')
      close (unit=2,status='save')
C
      write(6,300) char(7),char(7),char(7)
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C XINT1
C
C Ecrit par : Luc Cournoyer                                  Avril 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme d'interpolation lineaire dans l'axe X d'un fichier
C       deja corrige par XCOR. Les donnees sont interpolees sur une
C       grille reguliere pour permettre un affichage sur ecran graphique.
C
C       Version ou les valeurs minimums et maximums de l'interpolation
C       sont demandes a l'utilisateur
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*4 lines, columns, kx, norg
        real*4 scl, xof, yof, grdnull
        dimension vecin(1024,1124),zout(1024)
C
C Liste des formats
30      format (1a80)
250     format ('+','Interpolation de la ',I4,' ieme ligne')
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format (' La dimension des pixels en X est de ',F10.5,' unite(s)')
280     format (' L''affichage en X se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
290     format (' La dimension des pixels en Y est de ',F10.5,' unite(s)')
300     format (3a1)
C
        type*,' '
        type*,'      Interpolation lineaire en "X" pour l''affichage'
        type*,' '
        type*,'Quel est le nom du fichier (X,Z) a interpoler ?'
        read(5,30) infile
        type*,' '
        open(unit=1,name=infile//'.XCO',type='old',access='sequential',
     &  form='unformatted')
        read (1) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,xmin,xmax
        open(unit=2,name=infile//'.XIN',type='new',access='sequential',
     &  form='unformatted')
C
C Determination des maximums et mimimums dans l'axe des "X"
        type*,' '
        type*,'Dimensions de l''image (x,y) : ',columns,lines
        type*,'Quel sont les limites de l''interpolation (xmin,xmax) ? '
        read*,xmin,xmax
        type*,'Lecture du fichier et determination des extremums'
        type*,' '
        type*,' '
```

```
      do i=1,lines
            read(1) (vecin(i,k),k=1,2*columns)
      enddo
C
C Determination de l'increment
      xinc = (xmax-xmin)/columns
      write(6,260) scl
      write(6,290) yinc
      write(6,270) xinc
      write(6,280) xmin,xmax
      kx = 0
      norg = 0
      xof = 0.0
      yof = 0.0
      grdnull = -99990.
      write (2) columns,lines,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,xmin,xmax,(xi,i=13,126)
C
      type*,' '
      type*,' '
      type*,' '
C
C Interpolation lineaire
      do i=1,lines
            write (6,250) i
C On trouve la premiere valeur valide de la ligne
            k = 1
            do while (vecin(i,k).eq.0.and.vecin(i,k+1).eq.14000)
                  k = k + 2
            enddo
            inival = k
C Depart de la boucle d'interpolation pour la ligne
            kd = inival
            x = xmin
            do j=1,columns
                  k = kd
                  if (x.lt.vecin(i,inival)) then
                        zout(j) = 0.0
                  else
                        do while (vecin(i,k).lt.x)
                              kd = k
                              k = k + 2
                              if (k.gt.2*columns) then
                                    zout(j) = 0.0
                                    goto 999
                              endif
                              do while (vecin(i,k).eq.0)
                                    k = k + 2
                                    if (k.gt.2*columns) then
                                          zout(j) = 0.0
                                          goto 999
                                    endif
                              enddo
                        enddo
                        if (k.eq.inival) then
                              zout(j) = vecin(i,k+1)
                        else
```

```
                    slope = (vecin(i,k+1)-vecin(i,kd+1))/(vecin(i,k)-vecin(i,kd))
                    zout(j) = (slope*x) + vecin(i,k+1) - (slope*vecin(i,k))
                              endif
                         endif
c                        write(5,1) j,kd,k
c                        write(5,2) x,vecin(i,kd),vecin(i,k)
c                        write(5,2) zout(j),vecin(i,kd+1),vecin(i,k+1)
999                      x = x + xinc
                  enddo
                  write (2) (zout(m),m=1,columns)
         enddo
1        format(3I12)
2        format(3f12.4)
C
C Fermeture des fichiers
         close (unit=1,status='keep')
         close (unit=2,status='save')
C
         write(6,300) char(7),char(7),char(7)
C
         end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C YINT
C
C Ecrit par : Luc Cournoyer                                  Mars 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme d'interpolation lineaire dans l'axe Y d'un fichier
C       deja corrige par XCOR et interpole par XINT. Les donnees sont
C       interpolees sur une grille reguliere pour permettre un affichage
C       sur ecran graphique. Les bornes de l'interpolation sont les
C       minimums et maximums rencontres lors de la correction.
C
C       Le programme effectue la correction dans l'axe Y avant de
C       faire l'interpolation. Cette correction est faite de la meme
C       facon que dans le programme YCOR, soit avec les parametres
C       optimises lors de la calibration et d'une relation trigonometrique
C       simple.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile,heure
        integer row,col
        real zin(512,512),imaz(512,512),imay(512,512),zout(512,512)
        real phicos(512),phisin(512),dely,pi,ymin,ymax,y,phistart
C
C Liste des formats
30      format (1a80)
240     format ('0','Correction des donnees en Y ')
250     format ('+','Interpolation de la ',I3,' ieme colonne')
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format ('0','La dimension des pixels en X est de ',
     &  F10.5,' unite(s)')
280     format (' L''affichage en X se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
290     format ('0','La dimension des pixels en Y est de ',
     &  F10.5,' unite(s)')
300     format (' L''affichage en Y se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
C
C Lecture du fichier a corriger pour l'affichage
C
        type*,' '
        type*,'      Interpolation lineaire en "Y" pour l''affichage'
        type*,' '
        type*,'Quel est le nom du fichier a interpoler ?'
        read(*,30) infile
        type*,' '
        open(unit=1,name=infile//'.XIN',type='old',access='sequential',
     &  form='unformatted')
        read (1) col,row,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax
```

```
      open(unit=2,name=infile//'.YIN',type='new',access='sequential',
     & form='unformatted')
      type*,'Parametres a partir d''un fichier (1=oui) ?'
      read*,para
      if(para.eq.1) then
            type*,'Quel est le nom du fichier de parametres optimises ?'
            read(*,30) infile
            open(unit=3,name=infile//'.par',type='old',access='sequential',
     &      form='unformatted')
            read(3) h,phistart,delphi
            close (unit=3,status='keep')
            type*,' '
            type*,'Distance H = ',h
            type*,'Angle Phi de depart = ',phistart/(4.0*atan(1.0)/180.0)
            type*,'Increment angulaire = ',delphi/(4.0*atan(1.0)/180.0)
      else
            type*,' '
            type*,'Valeur de la distance H en unites ?'
            read*,h
            type*,'Valeur de l''angle de depart ?'
            read*,phistart
            phistart = phistart*4.0*atan(1.0)/180.0
            type*,'Valeur de l''increment angulaire ?'
            read*,delphi
            delphi = delphi*4.0*atan(1.0)/180.0
      endif
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
      do i=1,row
            phicos(i) = cos(phistart + (i-1)*delphi)
            phisin(i) = sin(phistart + (i-1)*delphi)
      enddo
C
C  Calibre l'image dans la direction "Y"
C
      write (*,240)
      do i=1,row
            read(1)(zin(i,j),j=1,col)
            do j=1,col
                  imay(i,j) = (h-zin(i,j)/scl)*phisin(i)
                  if (i.eq.1.and.j.eq.1) then
                        ymin = imay(i,j)
                        ymax = imay(i,j)
                  endif
                  if (imay(i,j).lt.ymin) ymin=imay(i,j)
                  if (imay(i,j).gt.ymax) ymax=imay(i,j)
                  imaz(i,j) = h - (h-zin(i,j)/scl)*phicos(i)
            enddo
      enddo
      close (unit=1,status='save')
C
      type*,' '
      type*,'Dimensions (x,y) : ',col,row
      yinc = (ymax-ymin)/row
```

```
      write(*,260) scl
      write(*,270) xinc
      write(*,280) xmin,xmax
      write(*,290) yinc
      write(*,300) ymin,ymax
      type*,' '
      type*,' '
C
C Interpolation lineaire pour obtenir un fichier uniforme pour l'affichage
C
      do i=1,col
            write (*,250) i
            k = 1
            inival = k
            kd = inival
            y = ymin
            do j=1,row
                  k = kd
                  if (y.lt.imay(inival,i)) then
                        zout(j,i) = 0.0
                  else
                        do while (imay(k,i).lt.y)
                              k = k + 1
                              if (k.gt.row) then
                                    zout(j,i) = 0.0
                                    goto 999
                              endif
                        enddo
                  slope=(imaz(k,i)-imaz(k-1,i))/(imay(k,i)-imay(k-1,i))
                  zout(j,i)=(slope*y) + imaz(k,i) - (slope*imay(k,i))
                  endif
999               y = y + yinc
            enddo
      enddo
C
C Ecriture de la nouvelle image dans le fichier de sortie
      type*,' '
      type*,' '
      type*,'Ecriture dans le fichier de sortie'
      write (2) col,row,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax,ymin,ymax,(i1,i=15,126)
      do j=1,row
            write (2) (zout(j,m)*scl,m=1,col)
      enddo
      close (unit=2,status='save')
C
      write(*,500) char(7),char(7),char(7)
500   format (3a1)
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C YINT1
C
C Ecrit par : Luc Cournoyer                                  Juin 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme d'interpolation lineaire dans l'axe Y d'un fichier
C       deja corrige par XCOR et interpole par XINT. Les donnees sont
C       interpolees sur une grille reguliere pour permettre un affichage
C       sur ecran graphique.
C       Version ou les valeurs minimums et maximums de l'interpolation
C       sont demandes a l'utilisateur.
C
C       Le programme effectue la correction dans l'axe Y avant de
C       faire l'interpolation. Cette correction est faite de la meme
C       facon que dans le programme YCOR, soit avec les parametres
C       optimises lors de la calibration et d'une relation trigonometrique
C       simple.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile,heure
        integer row,col
        real zin(512,512),imaz(512,512),imay(512,512),zout(512,512)
        real phicos(512),phisin(512),dely,pi,ymin,ymax,y,phistart
C
C Liste des formats
30      format (1a80)
240     format ('0','Correction des donnees en Y ')
250     format ('+','Interpolation de la ',I3,' ieme colonne')
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format ('0','La dimension des pixels en X est de ',
     &  F10.5,' unite(s)')
280     format (' L''affichage en X se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
290     format ('0','La dimension des pixels en Y est de ',
     &  F10.5,' unite(s)')
300     format (' L''affichage en Y se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
C
C Lecture du fichier a corriger pour l'affichage
C
        type*,' '
        type*,'      Interpolation lineaire en "Y" pour l''affichage'
        type*,' '
        type*,'Quel est le nom du fichier a interpoler ?'
        read(*,30) infile
        type*,' '
        open(unit=1,name=infile//'.XIN',type='old',access='sequential',
     &  form='unformatted')
        read (1) col,row,scl,kx,norg,xof,yof,grdnull
```

```
     &  ,xinc,yinc,xmin,xmax
        open(unit=2,name=infile//'.YIN',type='new',access='sequential',
     &  form='unformatted')
        type*,'Quel sont les limites de l''interpolation (ymin,ymax) ? '
        read*,ymin,ymax
        type*,'Parametres a partir d''un fichier (1=oui) ?'
        read*,para
        if(para.eq.1) then
                type*,'Quel est le nom du fichier de parametres optimises ?'
                read(*,30) infile
                open(unit=3,name=infile//'.par',type='old',access='sequential',
     &          form='unformatted')
                read(3) h,phistart,delphi
                close (unit=3,status='keep')
                type*,' '
                type*,'Distance H = ',h
                type*,'Angle Phi de depart = ',phistart/(4.0*atan(1.0)/180.0)
                type*,'Increment angulaire = ',delphi/(4.0*atan(1.0)/180.0)
        else
                type*,' '
                type*,'Valeur de la distance H en unites ?'
                read*,h
                type*,'Valeur de l''angle de depart ?'
                read*,phistart
                phistart = phistart*4.0*atan(1.0)/180.0
                type*,'Valeur de l''increment angulaire ?'
                read*,delphi
                delphi = delphi*4.0*atan(1.0)/180.0
        endif
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
        do i=1,row
                phicos(i) = cos(phistart + (i-1)*delphi)
                phisin(i) = sin(phistart + (i-1)*delphi)
        enddo
C
C  Calibre l'image dans la direction "Y"
C
        write (*,240)
        do i=1,row
                read(1)(zin(i,j),j=1,col)
                do j=1,col
                        imay(i,j) = (h-zin(i,j)/scl)*phisin(i)
                        imaz(i,j) = h - (h-zin(i,j)/scl)*phicos(i)
                enddo
        enddo
        close (unit=1,status='save')
C
        type*,' '
        type*,'Dimensions (x,y) : ',col,row
        yinc = (ymax-ymin)/row
        write(*,260) scl
        write(*,270) xinc
        write(*,280) xmin,xmax
```

```
      write(*,290) yinc
      write(*,300) ymin,ymax
      type*,' '
      type*,' '
C
C Interpolation lineaire pour obtenir un fichier uniforme pour l'affichage
C
      do i=1,col
            write (*,250) i
            k = 1
            inival = k
            kd = inival
            y = ymin
            do j=1,row
                  k = kd
                  if (y.lt.imay(inival,i)) then
                        zout(j,i) = 0.0
                  else
                        do while (imay(k,i).lt.y)
                              k = k + 1
                              if (k.gt.row) then
                                    zout(j,i) = 0.0
                                    goto 999
                              endif
                        enddo
                  slope=(imaz(k,i)-imaz(k-1,i))/(imay(k,i)-imay(k-1,i))
                  zout(j,i)=(slope*y) + imaz(k,i) - (slope*imay(k,i))
                  endif
999               y = y + yinc
            enddo
      enddo
C
C Ecriture de la nouvelle image dans le fichier de sortie
      type*,' '
      type*,' '
      type*,'Ecriture dans le fichier de sortie'
      write (2) col,row,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax,ymin,ymax,(i1,i=15,126)
      do j=1,row
            write (2) (zout(j,m)*scl,m=1,col)
      enddo
      close (unit=2,status='save')
C
      write(*,500) char(7),char(7),char(7)
500   format (3a1)
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C CYLCOR
C
C Ecrit par : Luc Cournoyer                                   Aout 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de correction de donnees prisent avec rotation de
C       l'objet. La position de l'axe de rotation est soustraite des
C       donnees mesurees et la correction est effectuee en fonction de
C       l'angle de la table tournante. Il faut donc connaitre la position
C       de l'axe de rotation en unites par rapport au images de calibration
C       et l'angle de la table tournante entre deux lignes de balayage.
C
C       Le fichier d'entree est celui corrige en X et Z par le programme
C       XCOR. Le format du fichier de sortie est compatible avec l'editeur
C       de solide de la section d'infographie. Ce format consiste en un
C       registre d'information sur les dimensions et les facteurs d'echelle,
C       suivis de trois tableaux X, Y, et Z. Les dimensions de ces tableaux
C       sont les memes que celles de l'image originale et les coordonnees X,
C       Y et Z des points non valides sont remplaces par la valeur de zero.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      integer row,col
      real zin(1024),imax(512,512),imay(512,512),imaz(512,512)
      real*8 h,delalpha
      character*80 infile
      real phi(512),phicos(512),phisin(512),phicos1(512),phisin1(512)
      real dely,ymin,ymax,y
C
C Liste des formats
30    format (1a80)
240   format ('+','Correction pour la ',I4,' ieme ligne soit a '
     &  ,F6.2,' degres')
260   format ('0','Le facteur d''echelle en Z est de ',F10.4)
C
C Lecture du fichier a corriger pour l'affichage
C
      type*,' '
      type*,'                    Correction de donnees cylindriques'
      type*,' '
      type*,'Quel est le nom du fichier (X,Z) a corriger ?'
      read(5,30)infile
      open(unit=1,name=infile//'.XCO',type='old',access='sequential',
     &  form='unformatted')
      read (1) col,row,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax
      open(unit=2,name=infile//'.CCO',type='new',access='sequential',
     &  form='unformatted')
C
C Lecture des parametres de calibration
```

```
      type*,'Quel est l''angle de depart de la table tournante (degre) ?'
      read*,depart
      type*,'Quel est l''increment angulaire entre deux lignes (degre) ?'
      read*,delalpha
      type*,'Quel est la position en Z de l''axe de rotation (cm) ?'
      read*,ref
      type*,' '
C
      type*,'Dimensions (x,y) : ',col,row
      write(6,260) scl
      type*,' '
      type*,' '
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
      do i=1,row
            phi(i) = depart + (i-1)*delalpha
            phicos(i) = cos(phi(i)*4*atan(1.0)/180.0)
            phisin(i) = sin(phi(i)*4*atan(1.0)/180.0)
            phicos1(i) = cos(phi(i)*4*atan(1.0)/180.0 + 4*atan(1.0))
            phisin1(i) = sin(phi(i)*4*atan(1.0)/180.0 + 4*atan(1.0))
      enddo
C
C  Correction de l'image dans la direction "Y"
C
      do i=1,row
            write (6,240) i,phi(i)
            read(1)(zin(j),j=1,2*col)
            do j=1,col
                  if (zin(2*j).ne.0) then
                        v = v + 1.0
                        imax(i,j) = zin(2*j-1)
                        if (zin(2*j).ge.ref) then
                        imay(i,j) = (zin(2*j)/scl-ref)*phicos(i)
                        imaz(i,j) = (zin(2*j)/scl-ref)*phisin(i)
                        else
                        imay(i,j) = (ref-zin(2*j)/scl)*phicos1(i)
                        imaz(i,j) = (ref-zin(2*j)/scl)*phisin1(i)
                        endif
                  else
                        imax(i,j) = 0.0
                        imay(i,j) = 0.0
                        imaz(i,j) = 0.0
                  endif
            enddo
      enddo
C
      type*,'Le nombre de points valides est de ',v,' sur ',col*row
      type*,'soit ',(v*100.0)/(col*row),' % des points'
      type*,' '
      type*,'Ecriture des donnees dans le fichier de sortie'
      write (2) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
      write (2) ((imax(i,j),j=1,col),i=1,row)
      write (2) ((imay(i,j),j=1,col),i=1,row)
      write (2) ((imaz(i,j),j=1,col),i=1,row)
```

```
      write(6,500) char(7),char(7),char(7)
      type*,'Fichiers DPICT X,Y et Z (1=oui) ?'
      read*,dpict
      if (dpict.eq.1) then
            open(unit=3,name=infile//'.CYX',type='new',
     &      access='sequential',form='unformatted')
            write (3) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
            open(unit=4,name=infile//'.CYY',type='new',
     &      access='sequential',form='unformatted')
            write (4) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
            open(unit=5,name=infile//'.CYZ',type='new',
     &      access='sequential',form='unformatted')
            write (5) col,row,scl,kx,norg,xof,yof,grdnull,(i1,i=9,126)
            do i=1,row
            write (3) ((imax(i,j),j=1,col))
            write (4) ((imay(i,j),j=1,col))
            write (5) ((imaz(i,j),j=1,col))
            enddo
      close (unit=3,status='save')
      close (unit=4,status='save')
      close (unit=5,status='save')
      endif
C
      close (unit=1,status='keep')
      close (unit=2,status='save')
C
      write(6,500) char(7),char(7),char(7)
500   format (3a1)
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C CYLCOR1
C
C Ecrit par : Luc Cournoyer                                   Aout 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de correction de donnees prisent avec rotation de
C       l'objet. La position de l'axe de rotation est soustraite des
C       donnees mesurees et la correction est effectuee en fonction de
C       l'angle de la table tournante. Il faut donc connaitre la position
C       de l'axe de rotation en unites par rapport aux images de calibration
C       et l'angle de latable tournante entre deux lignes de balayage.
C
C       Le fichier d'entree est celui corrige en X et Z par le programme
C       XCOR. Le fichier produit est de format DPICT BINAIRE XYZ ou ASCII
C       en raison d'un triplet XYZ par ligne.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        integer row,col
        real zin(1024),imaz(512),imay(512)
        real*8 h,delalpha
        character*80 infile
        real phi(512),phicos(512),phisin(512),phicos1(512),phisin1(512)
        real dely,ymin,ymax,y
C
C Liste des formats
30      format (1a80)
240     format ('+','Correction pour la ',I4,' ieme ligne soit a '
     &  ,F6.2,' degres')
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format ('0','La dimension des pixels en X est de ',
     &  F10.5,' unite(s)')
280     format (' L''affichage en X se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
300     format(1X,3(F7.2))
C
C Lecture du fichier a corriger pour l'affichage
C
        type*,' '
        type*,'                    Correction de donnees cylindriques'
        type*,' '
        type*,'Quel est le nom du fichier (X,Z) a corriger ?'
        read(5,30)infile
        open(unit=1,name=infile//'.XCO',type='old',access='sequential',
     &  form='unformatted')
        read (1) col,row,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax
C
C Lecture des parametres de calibration
        type*,'Quel est l''angle de depart de la table tournante (degre) ?'
```

```
      read*,depart
      type*,'Quel est l''increment angulaire entre deux lignes (degre) ?'
      read*,delalpha
      type*,'Quel est la position en Z de l''axe de rotation (cm) ?'
      read*,ref
      type*,' '
C
      type*,'Dimensions (x,y) : ',col,row
      yinc = (ymax-ymin)/row
      write(6,260) scl
      write(6,270) xinc
      write(6,280) xmin,xmax
      type*,' '
      type*,' '
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
      do i=1,row
            phi(i) = depart + (i-1)*delalpha
            phicos(i) = cos(phi(i)*4*atan(1.0)/180.0)
            phisin(i) = sin(phi(i)*4*atan(1.0)/180.0)
            phicos1(i) = cos(phi(i)*4*atan(1.0)/180.0 + 4*atan(1.0))
            phisin1(i) = sin(phi(i)*4*atan(1.0)/180.0 + 4*atan(1.0))
      enddo
C
C  Correction de l'image dans la direction "Y"
C
      type*,'Ecriture binaire (1) ou ASCII (2) ?'
      read*,ecrit
      if (ecrit.eq.1) then
      open(unit=2,name=infile//'.CYB',type='new',access='sequential',
     & form='unformatted')
      do i=1,row
            write (6,240) i,phi(i)
            read(1)(zin(j),j=1,2*col)
            do j=1,col
                  if (zin(2*j).ne.0) then
                        v = v + 1.0
                        if (zin(2*j).ge.ref) then
                        imay(j) = (zin(2*j)/scl-ref)*phicos(i)
                        imaz(j) = (zin(2*j)/scl-ref)*phisin(i)
                        else
                        imay(j) = (ref-zin(2*j)/scl)*phicos1(i)
                        imaz(j) = (ref-zin(2*j)/scl)*phisin1(i)
                        endif
                        write (2) zin(2*j-1),imay(j),imaz(j)
                  endif
            enddo
      enddo
      else
      open(unit=2,name=infile//'.CYA',type='new',form='formatted')
      do i=1,row
            write (6,240) i,phi(i)
            read(1)(zin(j),j=1,2*col)
            do j=1,col
```

```
                        if (zin(2*j).ne.0) then
                                v = v + 1.0
                                if (zin(2*j).ge.ref) then
                                imay(j) = (zin(2*j)/scl-ref)*phicos(i)
                                imaz(j) = (zin(2*j)/scl-ref)*phisin(i)
                                else
                                imay(j) = (ref-zin(2*j)/scl)*phicos1(i)
                                imaz(j) = (ref-zin(2*j)/scl)*phisin1(i)
                                endif
                                write (2,300) zin(2*j-1),imay(j),imaz(j)
                        endif
                enddo
        enddo
        endif
C
C       close (unit=1,status='keep')
        close (unit=2,status='save')
C
        write(6,500) char(7),char(7),char(7)
500     format (3a1)
C
        end
```

ANNEXE D

Sources des programmes de mesures sur VAX-11/750

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C CARMES
C
C Ecrit par : Luc Cournoyer                                   Mai 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de mesure de distances 3D a partir d'une image
C       originale et des coordonnees des points a mesurer. Les points
C       sont corriges en X,Y et Z avant que soient prises les mesures.
C
C       Un fichier .CAM est cree pour contenir les valeurs 3D de
C       chaque point ainsi que les distances qui les separent.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,parafile
        integer*2 buffer,p(1024)
        integer lines, columns, kx, norg
        real*4 scl, xof, yof, grdnull
        real phi(512),phicos(512),phisin(512),dely
        dimension xxx(100),yyy(100),zzz(100),px(100),py(100)
        dimension tx1(100),ty1(100),c1(200),tx2(100),ty2(100),c2(200)
        common /option/no(30),fp0
        data ir,ip/10,6/
C
C Liste des formats
30      format (1a80)
47      format ('0','                    List of measurements')
48      format ('0','Measurement betwen ',I2,' points')
49      format ('0')
50      format (' Point # ',I2,'   x = ',F8.4,
     &  '   y = ',F8.4,'   z = ',F8.4,'   px = ',I3,'   py = ',I3)
60      format (' Point # ',I2,'   *** DONNEE NON VALIDE ***
     &                    px = ',I3,'   py = ',I3)
61      format (' Point # ',I2,'   *** INVALID DATA FOR THIS POINT ***
     &          px = ',I3,'   py = ',I3)
100     format (' Distance entre les points',I3,
     &  ' et',I3,' de ',F8.4,' unite(s)')
101     format (' The distance betwen the points',I3,
     &  ' and',I3,' is',F8.4,' unit(s)')
C
        type*,' '
        type*,'                 Mesures de coordonnees 3D'
        type*,'                 (coordonnees cartesiennes)'
        type*,' '
C
        type*,'Quel fichier est a utiliser ?'
        read(*,30) infile
        open(unit=3,name=infile,type='old',access='sequential',
     &  form='unformatted')
        type*,'Quel sont les dimensions (x,y) ?'
```

```
      read*,columns,lines
C
C Lecture des parametres de calibration
      type*,'Parametres a partir d''un fichier (1=oui) ?'
      read*,para
      if(para.eq.1) then
            type*,'Quel est le nom du fichier de parametres optimises ?'
            read(*,30) parafile
            open(unit=1,name=parafile,type='old',access='sequential',
     &      form='unformatted')
            read(1) h,phistart,delphi
            close (unit=1,status='keep')
            type*,'Distance H = ',h
            type*,'Angle Phi de depart = ',phistart/(4.0*atan(1.0)/180.0)
            type*,'Increment angulaire = ',delphi/(4.0*atan(1.0)/180.0)
            type*,' '
      else
            type*,' '
            type*,'Valeur de la distance H en unites ?'
            read*,h
            type*,'Valeur de l''angle de depart ?'
            read*,phistart
            phistart = phistart*4.0*atan(1.0)/180.0
            type*,'Valeur de l''increment angulaire ?'
            read*,delphi
            delphi = delphi*4.0*atan(1.0)/180.0
            type*,' '
      endif
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
      do i=1,lines
            phicos(i) = cos(phistart + (i-1)*delphi)
            phisin(i) = sin(phistart + (i-1)*delphi)
      enddo
C
C Lecture des parametres de la fonction SPLINE
      type*,'Quel fichier contient les parametres de correction "Z" ?'
      read(*,30) parafile
      open(unit=1,name=parafile//'.PAR',type='old',access='sequential',
     & form='unformatted')
      read (1) nx1,ny1,kx1,ky1,xb1,xe1,yb1,ye1,sc1
      read (1) (tx1(i),i=1,nx1)
      read (1) (ty1(i),i=1,ny1)
      read (1) (c1(i),i=1,(nx1-kx1-1)*(ny1-ky1-1))
      close (unit=1,status='keep')
C
      type*,'Quel fichier contient les parametres de correction "X" ?'
      read(*,30) parafile
      open(unit=2,name=parafile//'.PAR',type='old',access='sequential',
     & form='unformatted')
      read (2) nx2,ny2,kx2,ky2,xb2,xe2,yb2,ye2
      read (2) (tx2(i),i=1,nx2)
      read (2) (ty2(i),i=1,ny2)
      read (2) (c2(i),i=1,(nx2-kx2-1)*(ny2-ky2-1))
```

```
      close (unit=2,status='keep')
C
C Correction des donnees a partir des parametres
C
      type*,' '
      type*,'Combien de points a mesurer (-1=mesure) ?'
      read*,boucle
500   open(unit=4,name=infile//'.CAM',type='new',form='formatted')
      do while (boucle.ne.-1)
      write(4,47)
      type*,' '
      type*,'Donnez les coordonnees Px et Py de chaque point :'
      type*,' '
      nbpoints = boucle
      write (4,48) nbpoints
      write (4,49)
      do i=1,nbpoints
            type*,'Point #',i
            read*,px(i),py(i)
      enddo
C
      inc = 1
      do while (inc.le.nbpoints)
      rewind (3)
      do i=1,lines
            read (3) (p(k),k=1,columns)
            if (i.eq.py(inc)) then
            do j=1,columns
                  if (j.eq.px(inc)) then
                  call mvbits(p(j),14,1,ibuf,0)
                  if (ibuf.eq.1) then
                        call mvbits(p(j),0,13,buffer,0)
                        x = j
                        y = buffer
                        zz = bisp(tx1,nx1,ty1,ny1,kx1,ky1,c1,x,y)
                        xxx(inc) = bisp(tx2,nx2,ty2,ny2,kx2,ky2,c2,x,y)
                        yyy(inc) = (h-zz)*phisin(i)
                        zzz(inc) = h-(h-zz)*phicos(i)
                        write(*,50) inc,xxx(inc),yyy(inc),zzz(inc),j,i
                        write(4,50) inc,xxx(inc),yyy(inc),zzz(inc),j,i
                  else
                        xxx(inc) = 0.0
                        yyy(inc) = 0.0
                        zzz(inc) = 0.0
                        write(*,60) inc,j,i
                        write(4,61) inc,j,i
                  endif
                  endif
            enddo
            endif
      enddo
      inc = inc + 1
C Fin de la deuxieme boucle while
      enddo
C
      type*,' '
      type*,'Combien de points a mesurer (-1=mesure) ?'
```

```
        read*,boucle
        type*,' '
C
C Fin de la boucle WHILE
        enddo
C
C Ecriture des donnees corrigees
600     type*,' '
        if (nbpoints.ge.2) then
                do i=2,nbpoints
                if (xxx(i).ne.0.or.yyy(i).ne.0.or.zzz(i).ne.0) then
                        type*,' '
                        write (4,49)
                        do j=1,i-1
                        if (xxx(j).ne.0.or.yyy(j).ne.0.or.zzz(j).ne.0) then
                                range = SQRT((xxx(j)-xxx(i))**2 +
     &                                  (yyy(j)-yyy(i))**2+(zzz(j)-zzz(i))**2)
                                write (*,100) i,j,range
                                write (4,101) i,j,range
                        endif
                        enddo
                endif
                enddo
        else
                type*,'Pas de distance 3D mesurable avec un seul point'
        endif
        type*,' '
        type*,'Une autre lecture (# points), mesure (-1) ou sortir (0) ?'
        read*,boucle
        close (unit=4,status='save')
        if (boucle.eq.0) goto 999
        if (boucle.eq.-1) then
                goto 600
        else
                goto 500
        endif
C
C Fermeture des fichiers
999     close (unit=3,status='keep')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C CYLMES
C
C Ecrit par : Luc Cournoyer                                  Mai 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de mesure de distances 3D a partir d'une image
C       originale et des coordonnees des points a mesurer. Les points
C       sont corriges en X et Z et la transformation des donnees
C       cylindrique a cartesienne est effectuee avant que soient prises
C       les mesures.
C
C       Un fichier .CYM est cree pour contenir les valeurs 3D de
C       chaque point ainsi que les distances qui les separent.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,parafile
        integer*2 buffer,p(1024)
        integer lines, columns, kx, norg
        real*4 scl, xof, yof, grdnull
        real phi(512),phicos(512),phisin(512),phicos1(512),phisin1(512)
        real dely,ymin,ymax,y
        dimension xxx(100),yyy(100),zzz(100),px(100),py(100)
        dimension tx1(100),ty1(100),c1(200),tx2(100),ty2(100),c2(200)
        common /option/no(30),fp0
        data ir,ip/10,6/
C
C Liste des formats
30      format (1a80)
47      format ('0','                    List of measurements')
48      format ('0','Measurement betwen ',I2,' points')
49      format ('0')
50      format (' Point # ',I2,'   x = ',F8.4,
     &  '   y = ',F8.4,'   z = ',F8.4,'   px = ',I3,'   py = ',I3)
60      format (' Point # ',I2,'   *** DONNEE NON VALIDE ***
     &                      px = ',I3,'   py = ',I3)
61      format (' Point # ',I2,'   *** INVALID DATA FOR THIS POINT ***
     &           px = ',I3,'   py = ',I3)
100     format (' Distance entre les points',I3,
     &  ' et',I3,' de ',F8.4,' unite(s)')
101     format (' The distance betwen the points',I3,
     &  ' and',I3,' is',F8.4,' unit(s)')
C
        type*,' '
        type*,'                  Mesures de coordonnees 3D'
        type*,'                  (coordonnees cylindriques)'
        type*,' '
C
        type*,'Quel fichier est a utiliser ?'
        read(*,30) infile
```

```
      open(unit=3,name=infile,type='old',access='sequential',
     &  form='unformatted')
      type*,'Quel sont les dimensions (x,y) ?'
      read*,columns,lines
C
C Lecture des parametres de calibration
      type*,'Quel est l''angle de depart de la table tournante (degre) ?'
      read*,depart
      type*,'Quel est l''increment angulaire entre deux lignes (degre) ?'
      read*,delalpha
      type*,'Quel est la position en Z de l''axe de rotation (unites) ?'
      read*,ref
      type*,' '
C
C Genere les angles correspondant aux differentes lignes
C et effectue le calcul de leur cosinus pour utilisation
C ulterieure pour accelerer les calculs subsequents.
C
      do i=1,lines
            phi(i) = depart + (i-1)*delalpha
            phicos(i) = cos(phi(i)*4*atan(1.0)/180.0)
            phisin(i) = sin(phi(i)*4*atan(1.0)/180.0)
            phicos1(i) = cos(phi(i)*4*atan(1.0)/180.0 + 4*atan(1.0))
            phisin1(i) = sin(phi(i)*4*atan(1.0)/180.0 + 4*atan(1.0))
      enddo
C
C Lecture des parametres de la fonction SPLINE
      type*,'Quel fichier contient les parametres de correction "Z" ?'
      read(*,30) parafile
      open(unit=1,name=parafile//'.PAR',type='old',access='sequential',
     &  form='unformatted')
      read (1) nx1,ny1,kx1,ky1,xb1,xe1,yb1,ye1,sc1
      read (1) (tx1(i),i=1,nx1)
      read (1) (ty1(i),i=1,ny1)
      read (1) (c1(i),i=1,(nx1-kx1-1)*(ny1-ky1-1))
      close (unit=1,status='keep')
C
      type*,'Quel fichier contient les parametres de correction "X" ?'
      read(*,30) parafile
      open(unit=2,name=parafile//'.PAR',type='old',access='sequential',
     &  form='unformatted')
      read (2) nx2,ny2,kx2,ky2,xb2,xe2,yb2,ye2
      read (2) (tx2(i),i=1,nx2)
      read (2) (ty2(i),i=1,ny2)
      read (2) (c2(i),i=1,(nx2-kx2-1)*(ny2-ky2-1))
      close (unit=2,status='keep')
C
C Correction des donnees a partir des parametres
C
      type*,' '
      type*,'Combien de points a mesurer ?'
      read*,boucle
500   open(unit=4,name=infile//'.CYM',type='new',form='formatted')
      do while (boucle.ne.-1)
      write(4,47)
      type*,' '
      type*,'Donnez les coordonnees Px et Py de chaque point :'
```

```
        type*,' '
        nbpoints = boucle
        write (4,48) nbpoints
        write (4,49)
        do i=1,nbpoints
                type*,'Point #',i
                read*,px(i),py(i)
        enddo
C
        inc = 1
        do while (inc.le.nbpoints)
        rewind (3)
        do i=1,lines
                read (3) (p(k),k=1,columns)
                if (i.eq.py(inc)) then
                do j=1,columns
                        if (j.eq.px(inc)) then
                        call mvbits(p(j),14,1,ibuf,0)
                        if (ibuf.eq.1) then
                                call mvbits(p(j),0,13,buffer,0)
                                x = j
                                y = buffer
                                zz = bisp(tx1,nx1,ty1,ny1,kx1,ky1,c1,x,y)
                                xxx(inc) = bisp(tx2,nx2,ty2,ny2,kx2,ky2,c2,x,y)
                                if (zz.ge.ref) then
                                yyy(inc) = (zz-ref)*phicos(i)
                                zzz(inc) = (zz-ref)*phisin(i)
                                else
                                yyy(inc) = (zz-ref)*phicos1(i)
                                zzz(inc) = (zz-ref)*phisin1(i)
                                endif
                                write(*,50) inc,xxx(inc),yyy(inc),zzz(inc),j,i
                                write(4,50) inc,xxx(inc),yyy(inc),zzz(inc),j,i
                        else
                                xxx(inc) = 0.0
                                yyy(inc) = 0.0
                                zzz(inc) = 0.0
                                write(*,60) inc,j,i
                                write(4,61) inc,j,i
                        endif
                        endif
                enddo
                endif
        enddo
        inc = inc + 1
C Fin de la deuxieme boucle while
        enddo
C
        type*,' '
        type*,'Combien de points a mesurer (-1=mesure) ?'
        read*,boucle
        type*,' '
C
C Fin de la boucle WHILE
        enddo
C
C Ecriture des donnees corrigees
```

```
600     type*,' '
        if (nbpoints.ge.2) then
                do i=2,nbpoints
                if (xxx(i).ne.0.or.yyy(i).ne.0.or.zzz(i).ne.0) then
                        type*,' '
                        write (4,49)
                        do j=1,i-1
                        if (xxx(j).ne.0.or.yyy(j).ne.0.or.zzz(j).ne.0) then
                                range = SQRT((xxx(j)-xxx(i))**2 +
     &                                  (yyy(j)-yyy(i))**2+(zzz(j)-zzz(i))**2)
                                write (*,100) i,j,range
                                write (4,101) i,j,range
                        endif
                        enddo
                endif
                enddo
        else
                type*,'Pas de distance 3D mesurable avec un seul point'
        endif
        type*,' '
        type*,'Une autre lecture (# points), mesure (-1) ou sortir (0) ?'
        read*,boucle
        close (unit=4,status='save')
        if (boucle.eq.0) goto 999
        if (boucle.eq.-1) then
                goto 600
        else
                goto 500
        endif
C
C Fermeture des fichiers
999     close (unit=3,status='keep')
C
        end
```

ANNEXE E

Sources des programmes d'affichage graphique sur VAX-11/750

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C AFFI
C
C Ecrit par : Luc Cournoyer                                  Novembre 1985
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme general d'affichage sur ecran graphique et utilisant
C       le processeur d'image TRAPIX de RECOGNITION CONCEPTS INC. Le
C       programme utilise des sous-programmes de la librairie de TAU,
C       THEORY AND APPLICATIONS UNLIMITED CORPORATION.
C
C       Le progamme effectue de l'affichage en tons de gris et en
C       isometrie. On peut aussi effectuer des expansions (ZOOM) sur
C       l'ecran ou dans l'image meme. Les fichier utilisees sont ceux
C       provenant directement de la camera.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        IMPLICIT        INTEGER*2 (A-Z)
        CHARACTER*30    IMA_NAME,OPT,CANAL_OPT,AXE,SPEC_FILE
        INTEGER*4       IOS,KX,NORG,DELX,DELY
        REAL*4          GAINZ
C
15      format(1A30)
20      format(8I4)
21      format(7I4)
25      format(F5.3)
C
        SPEC_FILE='PARA.AFF'
C
C Ouverture du fichier de parametres
        OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='FORMATTED',IOSTAT=IOS)
        read(1,15) ima_name
        read(1,15) axe
        read(1,20) dimx,dimy,inx,iny,fix,fiy,depx,depy
        read(1,21) bitdep,canal,gainx,gainy,decal,interx,intery
        read(1,25) gainz
        close (unit=1,status='save')
C
        X1=DEPX
        Y1=DEPY
C
        SUCCESS = INIT()
        IF ( .NOT. SUCCESS ) CALL FATAL_ERROR( SUCCESS, ' ERROR: IN INIT ')
C
        efface=1
C
C Affichage du menu
C
100     type*, ' '
```

```
      type*, '                Programme d''affichage sur TRAPIX '
      type*, '                      (format TRAPIX)'
      type*, ' '
      type*, '        1. Nom du fichier image :              ',ima_name
      type*, '        2. Dimension de l''image (x,y) :  ',dimx,dimy
      type*, '        3. Position initiale :            ',inx,iny
      type*, '        4. Position finale :              ',fix,fiy
      type*, '        5. Position de depart :           ',depx,depy
      type*, '        6. Bit de depart (-1,0-8) :       ',bitdep
      type*, '        7. Canal (options specials) :     ',canal
      type*, ' '
      type*, '        8. Gains isometriques (x,y,z) :   ',gainx,gainy,gainz
      type*, '        9. Decalage des lignes (0=non) : ',decal
      type*, '       10. Tout les N pixels :            ',interx
      type*, '       11. Toute les M lignes :           ',intery
      type*, '       12. Axe du trace :                       ',axe
      type*, ' '
      type*, '        A. Affichage en niveau de gris '
      type*, '        B. Affichage en isometrique '
      type*, '        C. Parametres par defaut '
      type*, '        S. Sortie '
      type*,' '
      type*, 'Option : '
      read(5,15)opt
C
C Change les parametres dans le fichier exterieur, s'il y a lieu
C
      if (change.eq.1) then
       OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
      1 FORM='FORMATTED',IOSTAT=IOS)
      write(1,15) ima_name
      write(1,15) axe
      write(1,20) dimx,dimy,inx,iny,fix,fiy,depx,depy
      write(1,21) bitdep,canal,gainx,gainy,decal,interx,intery
      write(1,25) gainz
      close (unit=1,status='save')
      change=0
      endif
C
      if (opt.eq.'s'.or.opt.eq.'S') goto 1000
      if (opt.eq.'a'.or.opt.eq.'A') then
              call inten(SPEC_FILE)
      else
              if (opt.eq.'b'.or.opt.eq.'B') then
                      if (efface.eq.1) call zer(1,canal)
                      if (efface.eq.1) call cur(8,0,0,0,0,0,0,0,0,1)
                      call isoint(SPEC_FILE)
              else
                      if (opt.eq.'7') then
                      do while (opt.ne.'r'.and.opt.ne.'R')
                              type*,' '
                              type*,' D. Changement de canal '
                              type*,' E. Effacement de l''affichage canal '
                              type*,' F. Effacement (ou non) avant l''affichage'
                              type*,' O. Curseur pour origine '
                              type*,' X. Curseur pour dimensions '
                              type*,' Z. Zoom + scroll '
```

```
                type*,' '
                type*,' R. Retour au menu principal '
                type*,' '
                type*,' Option : '
                read(5,15) opt
                if (opt.eq.'d'.or.opt.eq.'D') then
                        type*,' '
                        type*,' Canal :'
                        read*,canal
                        call dis(1,canal)
                endif
                if (opt.eq.'e'.or.opt.eq.'E') then
                        call zer(1,canal)
                endif
                if (opt.eq.'f'.or.opt.eq.'F') efface=efface*-1
                if (opt.eq.'o'.or.opt.eq.'O') then
                call cur(8,1,depx,depy,depx,depy,4,0,0)
                        DNTCRE = RCR1(depx,depy)
                endif
                if (opt.eq.'x'.or.opt.eq.'X') then
                call cur(8,0,depx,depy,depx,depy,4,0,0)
                        DNTCRE = RCR1(X1,Y1)
                        DNTCRE = RCR2(X2,Y2)
                        fix=(X2-depx)/gainx + inx
                        fiy=(Y2-depy)/gainy + iny
                        inx=(X1-depx)/gainx + inx
                        iny=(Y1-depy)/gainy + iny
                        depx=X1
                        depy=Y1
                endif
              if (opt.eq.'z'.or.opt.eq.'Z') call scz(4,0,0,1,0)
        enddo           ! while loop end
        else
        type*,' '
        type*, 'Nouveau parametre : '
        type*,' '
        change=1
C
        if (opt.eq.'c'.or.opt.eq.'C') then
                inx=1
                iny=1
                fix=dimx
                fiy=dimy
                gainx=1
                gainy=1
                gainz=1.0
                depx=0
                depy=0
                bitdep=-1
                interx=1
                intery=1
                axe='deux'
                decal=1
        endif
C
        if (opt.eq.'12') then
                type*,'Axe HORizontal, VERtical ou les DEUX ? '
```

```
                        read(*,15) axe
                    endif
C
                    if (opt.eq.'1') read(*,15) ima_name
                    if (opt.eq.'2') read*, dimx,dimy
                    if (opt.eq.'3') then
                            read*, inx,iny
                            IF (inx.lt.1.or.iny.lt.1) THEN
                                    TYPE*,' '
            TYPE*,' *** LE POINT MINIMUM DE LA FENETRE EST (1,1) ***'
                                    inx=1
                                    iny=1
                            endif
                    endif
                    if (opt.eq.'4') then
                            read*, fix,fiy
                            IF (fix.gt.dimx.or.fiy.gt.dimy) THEN
                                    TYPE*,' '
            TYPE*,' *** LA FENETRE DEPASSE LA DIMENSION DE L''IMAGE ***'
                                    fix=dimx
                                    fiy=dimy
                            endif
                    endif
                    if (opt.eq.'5') read*, depx,depy
                    if (opt.eq.'6') read*, bitdep
                    if (opt.eq.'8') read*, gainx,gainy,gainz
                    if (opt.eq.'9') read*, decal
                    if (opt.eq.'10') read*, interx
                    if (opt.eq.'11') read*, intery
                    endif
             endif
        endif
C
C Lit les dimensions du fichier dans le fichier
C
         OPEN ( UNIT=1, NAME=IMA_NAME, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='UNFORMATTED',IOSTAT=IOS)
        CHANGE=1
        goto 100
C
1000    continue
C
        STATUS = RELESE()
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C AFFI1
C
C Ecrit par : Luc Cournoyer                                  Novembre 1985
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme general d'affichage sur ecran graphique et utilisant
C       le processeur d'image TRAPIX de RECOGNITION CONCEPTS INC. Le
C       programme utilise des sous-programmes de la librairie de TAU,
C       THEORY AND APPLICATIONS UNLIMITED CORPORATION.
C
C       Le progamme effectue de l'affichage en tons de gris ou en
C       isometrie. On peut aussi effectuer des expansions (ZOOM) sur
C       l'ecran ou dans l'image meme, ainsi que des mesures sur l'image.
C       Le progamme effectue de l'affichage en isometrie. On peut
C       aussi effectuer des expansions (ZOOM) sur l'ecran ou dans
C       l'image meme. Les fichier utilisees sont ceux dont le format
C       est du genre DPICT.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        IMPLICIT        INTEGER*2 (A-Z)
        CHARACTER*30    IMA_NAME,OPT,CANAL_OPT,AXE,SPEC_FILE
        INTEGER*4       IOS,KX,NORG,DELX,DELY
        real*4          GAINZ
C
15      format(1A30)
20      format(8I4)
21      format(7I4)
25      format(F5.3)
C
        SPEC_FILE='PARA.AFF'
C
C Ouverture du fichier de parametres
         OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='FORMATTED',IOSTAT=IOS)
        read(1,15) ima_name
        read(1,15) axe
        read(1,20) dimx,dimy,inx,iny,fix,fiy,depx,depy
        read(1,21) bitdep,canal,gainx,gainy,decal,interx,intery
        read(1,25) gainz
        close (unit=1,status='save')
C
        X1=DEPX
        Y1=DEPY
C
        SUCCESS = INIT()
        IF ( .NOT. SUCCESS ) CALL FATAL_ERROR( SUCCESS, ' ERROR: IN INIT ')
```

```
C
      efface=1
C
C Affichage du menu
C
100   type*, ' '
      type*, ' '
      type*, '                    Programme d''affichage sur TRAPIX '
      type*, '                          (format DPICT)'
      type*, ' '
      type*, '        1. Nom du fichier image :              ',ima_name
      type*, '        2. Dimension de l''image (x,y) :  ',dimx,dimy
      type*, '        3. Position initiale :            ',inx,iny
      type*, '        4. Position finale :              ',fix,fiy
      type*, '        5. Position de depart :           ',depx,depy
      type*, '        6. Bit de depart (-1,0-8) :       ',bitdep
      type*, '        7. Canal (options specials) :     ',canal
      type*, ' '
      type*, '        8. Gains isometriques (x,y,z) :   ',gainx,gainy,gainz
      type*, '        9. Decalage des lignes (0=non) : ',decal
      type*, '       10. Tout les N pixels :            ',interx
      type*, '       11. Toute les M lignes :           ',intery
      type*, '       12. Axe du trace :                       ',axe
      type*, ' '
      type*, '        A. Affichage en niveau de gris '
      type*, '        B. Affichage en isometrique '
      type*, '        C. Parametres par defaut '
      type*, '        S. Sortie '
      type*,' '
      type*, 'Option : '
      read(5,15)opt
C
C Change les parametres dans le fichier exterieur, s'il y a lieu
C
      if (change.eq.1) then
       OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
     1 FORM='FORMATTED',IOSTAT=IOS)
      write(1,15) ima_name
      write(1,15) axe
      write(1,20) dimx,dimy,inx,iny,fix,fiy,depx,depy
      write(1,21) bitdep,canal,gainx,gainy,decal,interx,intery
      write(1,25) gainz
      close (unit=1,status='save')
      change=0
      endif
C
      if (opt.eq.'s'.or.opt.eq.'S') goto 1000
      if (opt.eq.'a'.or.opt.eq.'A') then
              call intenl(SPEC_FILE)
              curaffi = 1
      else
      if (opt.eq.'b'.or.opt.eq.'B') then
              if (efface.eq.1) call zer(1,canal)
              if (efface.eq.1) call cur(8,0,0,0,0,0,0,0,1)
```

```
        call isoreal(SPEC_FILE)
else
        if (opt.eq.'7') then
        do while (opt.ne.'r'.and.opt.ne.'R')
                type*,' '
                type*,' D. Changement de canal '
                type*,' E. Effacement de l''affichage canal '
                type*,' F. Effacement (ou non) avant l''affichage'
                type*,' M. Affichage des mesures entre les curseurs'
                type*,' O. Curseur pour origine '
                type*,' X. Curseur pour dimensions '
                type*,' Z. Zoom + scroll '
                type*,' '
                type*,' R. Retour au menu principal '
                type*,' '
                type*,' Option : '
                read(5,15) opt
                if (opt.eq.'d'.or.opt.eq.'D') then
                        type*,' '
                        type*,' Canal :'
                        read*,canal
                        call dis(1,canal)
                endif
                if (opt.eq.'e'.or.opt.eq.'E') then
                        call zer(1,canal)
                endif
                if (opt.eq.'f'.or.opt.eq.'F') efface=efface*-1
                if (opt.eq.'m'.or.opt.eq.'M') then
                        call leccoor(ima_name,inx,fix,iny,fiy)
                endif
                if (opt.eq.'o'.or.opt.eq.'O') then
                call cur(8,1,X1,Y1,X1,Y1,1,0,0)
                        DNTCRE = RCR1(depx,depy)
                endif
                if (opt.eq.'x'.or.opt.eq.'X') then
                        if (curaffi.eq.1) then
                                call cur(8,2,depx,depy,depx,depy,6,0,0)
                                curaffi = 0
                        else
                                call cur(8,2,xx1,yy1,xx2,yy2.6,0,0)
                        endif
                        DNTCRE = RCR1(X1,Y1)
                        DNTCRE = RCR2(X2,Y2)
                        fix=(X2-depx)/gainx + inx
                        fiy=(Y2-depy)/gainy + iny
                        inx=(X1-depx)/gainx + inx
                        iny=(Y1-depy)/gainy + iny
                        depx=X1
                        depy=Y1
                        xx1 = x1
                        yy1 = y1
                        xx2 = x2
                        yy2 = y2
```

```
                endif
              if (opt.eq.'z'.or.opt.eq.'Z') call scz(4,0,0,1,0)
            enddo           ! while loop end
            else
            type*,' '
            type*, 'Nouveau parametre : '
            type*,' '
            change=1
C
            if (opt.eq.'c'.or.opt.eq.'C') then
                        inx=1
                        iny=1
                        fix=dimx
                        fiy=dimy
                        gainx=1
                        gainy=1
                        gainz=1.0
                        depx=0
                        depy=0
                        bitdep=-1
                        interx=1
                        intery=1
                        axe='deux'
                        decal=1
            endif
C
            if (opt.eq.'12') then
                    type*,'Axe HORizontal, VERtical ou les DEUX ? '
                    read(*,15) axe
            endif
C
            if (opt.eq.'1') read(*,15) ima_name
            if (opt.eq.'2') read*, dimx,dimy
            if (opt.eq.'3') then
                    read*, inx,iny
                    IF (inx.lt.1.or.iny.lt.1) THEN
                            TYPE*,' '
      TYPE*,' *** LE POINT MINIMUM DE LA FENETRE EST (1,1) ***'
                            inx=1
                            iny=1
                    endif
            endif
            if (opt.eq.'4') then
                    read*, fix,fiy
                    IF (FIX.GT.DELX.OR.FIY.GT.DELY) THEN
                            TYPE*,' '
      TYPE*,' *** LA FENETRE DEPASSE LA DIMENSION DE L''IMAGE ***'
                            fix=dimx
                            fiy=dimy
                    endif
            endif
            if (opt.eq.'5') read*, depx,depy
            if (opt.eq.'6') read*, bitdep
            if (opt.eq.'8') read*, gainx,gainy,gainz
```

```
                if (opt.eq.'9') read*, decal
                if (opt.eq.'10') read*, interx
                if (opt.eq.'11') read*, intery
                endif
        endif
        endif
C
C Lit les dimensions du fichier dans le fichier
C
         OPEN ( UNIT=1, NAME=IMA_NAME, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='UNFORMATTED',IOSTAT=IOS)
        READ(1) DELX,DELY
        DIMX=DELX
        DIMY=DELY
        CHANGE=1
        goto 100
C
1000    continue
C
        STATUS = RELESE()
C
        end
```

```
      DNTCRE  = WTRN(TRNSP)                    ! set up write transpose
C
C     open image file
C
       OPEN ( UNIT=1, NAME=STRING, TYPE='OLD', ACCESS='SEQUENTIAL',
      1 FORM='UNFORMATTED',IOSTAT=IOS)
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
C
      DO I=1,INIY-1
            READ(1)(FZI2(1))
      ENDDO
C
C
C FIND THE EXTREMUMS IF NEEDED
      IF (WIND.EQ.-1) THEN
            DO I=INIY,FINY,INTERY
                  DO K=1,INTERY
                        READ(1)(FZI2(J),J=1,FINX)
                  ENDDO
                  DO J=INIX,FINX
                        CALL MVBITS(FZI2(J),0,13,BUFFER,0)
                        IF (I.EQ.INIY.AND.J.EQ.INIX) THEN
                              ZMIN=BUFFER
                              ZMAX=BUFFER
                        ENDIF
                        ZMIN=IMIN0(ZMIN,BUFFER)
                        ZMAX=IMAX0(ZMAX,BUFFER)
                  ENDDO
            ENDDO
      ENDIF
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
C
      REWIND(1)
      DO I=1,INIY-1
            READ(1)(FZI2(1))
      ENDDO
C
C READ THE FILE AND WRITE TO THE TRAPIX
C
      INCY=Y0
      DO 150 I=INIY,FINY,INTERY
            DO 90 K=1,INTERY
            IF (I+K.GT.FINY+1) GOTO 150
            READ(1)(FZI2(J),J=1,FINX)
            CONTINUE
90
C
      DO 150 K=1,INTERY*FACTY
      INCX=X0
      DO 100 J=INIX,FINX,INTERX
C
C TAKE THE PROPER WINDOW OF 8 BITS
      IF (WIND.EQ.-1) THEN
            CALL MVBITS(FZI2(J),0,13,BUFFER,0)
            BUFFER=INT(255.0*(BUFFER-ZMIN)/(1.0*(ZMAX-ZMIN)))
      ELSE
```

```
            Z=FZI2(J)
            CALL MVBITS(Z,WIND,8,BUFFER,0)
        ENDIF
        DO 100 L=1,INTERX*FACTX
        DNTCRE=WP!X(INCX,INCY,BUFFER)              !SEND DATA TO TRAPIX
        INCX=INCX+1
100     CONTINUE
        INCY=INCY+1
150     CONTINUE
C
        CLOSE (UNIT=1,ERR=9989)
C
        RETURN
C
9989    CONTINUE
C
        STATUS=RELESE()
        CALL EXIT
C
        RETURN
C
        END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C INTEN1
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme d'ecriture graphique en tons de gris dans le
C       systeme TRAPIX, d'une image de format DPICT.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE INTEN1 (SPEC_FILE)
C
        IMPLICIT          INTEGER*2 (A-Z)
        CHARACTER*30      STRING,SPEC_FILE,AXE
        LOGICAL*4         SUCCESS
C
        INTEGER*4         IOS,KX,NORG,DELX,DELY
        REAL*4            FZR4(512),ZMAX,ZMIN,FACTZ
C
15      format(1A30)
20      format(8I4)
21      format(7I4)
25      format(F5.3)
C
C READ THE PARAMETERS
         OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='FORMATTED',IOSTAT=IOS)
        read(1,15) STRING
        read(1,15) axe
        read(1,20) dimx,dimy,INIX,INIY,FINX,FINY,X0,Y0
        read(1,21) WIND,CHANEL,FACTX,FACTY,SHIFT,INTERX,INTERY
        read(1,25) FACTZ
        close (unit=1,status='save')
C
        TRNSP = 0
        SUCCESS=INIT()
        IF(.NOT.SUCCESS)CALL FATAL_ERROR(SUCCESS,' ERROR : IN INIT')
C
        CALL FB2IDC( CHANEL,IDCNUM,CHAN,SCHN )  ! get user input
        DNTCRE  = WCHN(CHAN)                    ! set up write channel
        DNTCRE  = WSCH(SCHN)                    ! set up write sub-channel
        DNTCRE  = WTRN(TRNSP)                   ! set up write transpose
C
C       open image file
C
         OPEN ( UNIT=1, NAME=STRING, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='UNFORMATTED',IOSTAT=IOS)
C
C READ THE DIMENSIONS FROM THE STANDARD FILE OR THE PARAMETERS FILE
C
```

```
      READ(1) DELX,DELY
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
C
      DO I=1,INIY-1
            READ(1)(FZR4(1))
      ENDDO
C
C FIND THE EXTREMUMS IF NEEDED
      IF (WIND.EQ.-1) THEN
            DO I=INIY,FINY,INTERY
                  DO K=1,INTERY
                        READ(1)(FZR4(J),J=1,FINX)
                  ENDDO
                  DO J=INIX,FINX
                        IF (I.EQ.INIY.AND.J.EQ.INIX) THEN
                              ZMIN=FZR4(J)
                              ZMAX=FZR4(J)
                        ENDIF
                        ZMIN=AMIN1(ZMIN,FZR4(J))
                        ZMAX=AMAX1(ZMAX,FZR4(J))
                  ENDDO
            ENDDO
      ENDIF
C
C READ THE DIMENSIONS FROM THE STANDARD FILE OR THE PARAMETERS FILE
C
      REWIND(1)
      READ(1) DELX,DELY
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
C
      DO I=1,INIY-1
            READ(1)(FZR4(1))
      ENDDO
C
C READ THE FILE AND WRITE TO THE TRAPIX
C
      INCY=Y0
      DO 150 I=INIY,FINY,INTERY
            DO 90 K=1,INTERY
            IF (I+K.GT.FINY+1) GOTO 150
            READ(1)(FZR4(J),J=1,FINX)
90          CONTINUE
C
      DO 150 K=1,INTERY*FACTY
      INCX=X0
      DO 100 J=INIX,FINX,INTERX
C
C TAKE THE PROPER WINDOW OF 8 BITS
      IF (WIND.EQ.-1) THEN
            BUFFER = INT((255.0*(FZR4(J)-ZMIN))/(ZMAX-ZMIN))
      ELSE
            Z=INT(FZR4(J))
            CALL MVBITS(Z,WIND,8,BUFFER,0)
      ENDIF
      DO 100 L=1,INTERX*FACTX
```

```
      DNTCRE=WPIX(INCX,INCY,BUFFER)                !SEND DATA TO TRAPIX
      INCX=INCX+1
100   CONTINUE
      INCY=INCY+1
150   CONTINUE
C
      CLOSE (UNIT=1,ERR=9989)
C
      RETURN
C
9989  CONTINUE
C
      STATUS=RELESE()
      CALL EXIT
C
      RETURN
C
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C ISOINT
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme d'ecriture graphique en isometrie dans le
C       systeme TRAPIX, d'une image provenent de la camera directement.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE ISOINT (SPEC_FILE)
C
        IMPLICIT        INTEGER*2 (A-Z)
        CHARACTER*30    STRING,SPEC_FILE,AXE
        LOGICAL*4       SUCCESS
C
        PARAMETER       (NUM_PIX=512)
        INTEGER*2       PMAX(NUM_PIX),FTEMP(NUM_PIX)
        INTEGER*2       REGTEMP(NUM_PIX)
        INTEGER*4       IOS,KX,NORG
        INTEGER*2       FZ12(NUM_PIX),INTRACE(NUM_PIX)
        REAL*4          GAINZ
C
15      format(1A30)
20      format(8I4)
21      format(7I4)
25      format(F5.3)
C
C INITIALISE THE ARRAYS
        DO 30 A=1,NUM_PIX
        PMAX(A)=0
        REGTEMP(A)=0
        FTEMP(A)=0
        INTRACE(A)=100
30      CONTINUE
C
C READ THE PARAMETERS
         OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='FORMATTED',IOSTAT=IOS)
        read(1,15) STRING
        read(1,15) axe
        read(1,20) dimx,dimy,INIX,INIY,FINX,FINY,X0,Y0
        read(1,21) WIND,CHANEL,GAINX,GAINY,SHIFT,INTERX,INTERY
        read(1,25) GAINZ
        close (unit=1,status='save')
C
        TRNSP = 0
        SUCCESS=INIT()
        IF(.NOT.SUCCESS)CALL FATAL_ERROR(SUCCESS,' ERROR : IN INIT')
C
```

```
      CALL FB2IDC( CHANEL,IDCNUM,CHAN,SCHN )  ! get user input
      DNTCRE  = WCHN(CHAN)                    ! set up write channel
      DNTCRE  = WSCH(SCHN)                    ! set up write sub-channel
      DNTCRE  = WTRN(TRNSP)                   ! set up write transpose
C
C     open image file
C
       OPEN ( UNIT=1, NAME=STRING, TYPE='OLD', ACCESS='SEQUENTIAL',
     1 FORM='UNFORMATTED',IOSTAT=IOS)
C
C GENERATE THE IMAGE WITH THE PROPER FORMAT
C
      INCY=0
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
C
      DO I=1,INIY-1
            READ(1)(FZI2(1))
      ENDDO
C
C START OF THE LINE LOOP
C
      DO 150  I=INIY,FINY,INTERY
            DO 90 K=1,INTERY
                  IF (I+K.GT.FINY+1) GOTO 150
                        READ(1)(FZI2(J),J=1,FINX)
90          CONTINUE
C
C START OF THE PIXEL LOOP
C
      INCX=1
      DO 100  J=INIX,FINX,INTERX
C
C TAKE THE PROPER WINDOW OF 8 BITS
      Z=FZI2(J)
      CALL MVBITS(Z,WIND,8,BUFFER,0)
C
C EVALUATE THE MAXIMUM FOR THE GIVEN X POSITION
      FACTX=GAINX*INTERX
      FACTY=GAINY*INTERY
      NEWZ=Y0+BUFFER*GAINZ+INCY*FACTY
      IF (NEWZ.GT.REGTEMP(INCX)) THEN
            FMAX(INCX)=NEWZ
            INTRACE(INCX)=100
      ELSE
            FMAX(INCX)=REGTEMP(INCX)
            INTRACE(INCX)=IMIN0(250,INTRACE(INCX)+25)
      ENDIF
      X = X0 + INCX*FACTX + INCY*SHIFT*FACTX
C
C DRAW HORIZONTAL VECTORS IF WANTED
      IF(INCX-1.LT.1) GOTO 95
            IF(AXE.EQ.'DEUX'.OR.AXE.EQ.'deux'
     1      .OR.AXE.EQ.'HOR'.OR.AXE.EQ.'hor') THEN
      DNTCRE=VECTOR(X-FACTX,FMAX(INCX-1),X,FMAX(INCX),INTRACE(INCX),0)
            ENDIF
C
```

```
C DRAW VERTICAL VECTORS IF WANTED
95      IF (INCY.NE.0) THEN
                IF(AXE.EQ.'DEUX'.OR.AXE.EQ.'deux'
     1          .OR.AXE.EQ.'VER'.OR.AXE.EQ.'ver') THEN
        DNTCRE=VECTOR(X-SHIFT*FACTX,FTEMP(INCX),X,FMAX(INCX),INTRACE(INCX),0)
                ENDIF
        ENDIF
        INCX=INCX+1
100     CONTINUE
                DO 110 J=1,FINX
                FTEMP(J)=FMAX(J)
                IF (J+SHIFT.LT.1) GOTO 110
                REGTEMP(J)=FMAX(J+SHIFT)
110             CONTINUE
        INCY=INCY+1
150     CONTINUE
C
        CLOSE (UNIT=1,ERR=9989)
C
        RETURN
C
9989    CONTINUE
C
        STATUS=RELESE()
        CALL EXIT
C
        RETURN
C
        END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C ISOREAL
C
C Ecrit par : Luc Cournoyer                            Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme d'ecriture graphique en isometrie dans le
C       systeme TRAPIX, d'une image de format DPICT.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE ISOREAL (SPEC_FILE)
C
        IMPLICIT          INTEGER*2 (A-Z)
        CHARACTER*30      STRING,SPEC_FILE,AXE
        LOGICAL*4         SUCCESS
C
        PARAMETER         (NUM_PIX=512)
        INTEGER*2         FMAX(NUM_PIX),FTEMP(NUM_PIX)
        INTEGER*2         REGTEMP(NUM_PIX)
        INTEGER*4         IOS,KX,NORG,DELX,DELY
        REAL*4            FZR4(NUM_PIX),ZMAX,ZMIN,GAINZ
C
15      format(1A30)
20      format(8I4)
21      format(7I4)
25      format(1F5.3)
C
C INITIALISE THE ARRAYS
        DO 30 A=1,NUM_PIX
        FMAX(A)=0
        REGTEMP(A)=0
        FTEMP(A)=0
30      CONTINUE
C
C READ THE PARAMETERS
         OPEN ( UNIT=1, NAME=SPEC_FILE, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='FORMATTED',IOSTAT=IOS)
        read(1,15) STRING
        read(1,15) axe
        read(1,20) dimx,dimy,INIX,INIY,FINX,FINY,X0,Y0
        read(1,21) WIND,CHANEL,GAINX,GAINY,SHIFT,INTERX,INTERY
        read(1,25) GAINZ
        close (unit=1,status='save')
C
        IVAL = 150                           ! INTENSITY OF THE TRACE
        TRNSP = 0
        SUCCESS=INIT()
        IF(.NOT.SUCCESS)CALL FATAL_ERROR(SUCCESS,' ERROR : IN INIT')
C
        CALL FB2IDC( CHANEL,IDCNUM,CHAN,SCHN )  ! get user input
```

```
      DNTCRE  = WCHN(CHAN)                    ! set up write channel
      DNTCRE  = WSCH(SCHN)                    ! set up write sub-channel
      DNTCRE  = WTRN(TRNSP)                   ! set up write transpose
C
C     open image file
       OPEN ( UNIT=1, NAME=STRING, TYPE='OLD', ACCESS='SEQUENTIAL',
      1 FORM='UNFORMATTED',IOSTAT=IOS)
C
C READ THE DIMENSIONS FROM THE STANDARD FILE OR THE PARAMETERS FILE
      READ(1) DELX,DELY
C
C GENERATE THE IMAGE WITH THE PROPER FORMAT
      INCY=0
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
      DO I=1,INIY-1
            READ(1)(FZR4(1))
      ENDDO
C
C FIND THE EXTREMUMS IF NEEDED
      IF (WIND.EQ.-1) THEN
            DO I=INIY,FINY,INTERY
                  DO K=1,INTERY
                        READ(1)(FZR4(J),J=1,FINX)
                  ENDDO
                  DO J=INIX,FINX
                        IF (I.EQ.INIY.AND.J.EQ.INIX) THEN
                              ZMIN=FZR4(J)
                              ZMAX=FZR4(J)
                        ENDIF
                        ZMIN=AMIN1(ZMIN,FZR4(J))
                        ZMAX=AMAX1(ZMAX,FZR4(J))
                  ENDDO
            ENDDO
      ENDIF
C
C READ THE DIMENSIONS FROM THE STANDARD FILE OR THE PARAMETERS FILE
      REWIND(1)
      READ(1) DELX,DELY
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
      DO I=1,INIY-1
            READ(1)(FZR4(1))
      ENDDO
C
C START OF THE LINE LOOP
      DO 150  I=INIY,FINY,INTERY
            DO 90 K=1,INTERY
                  IF (I+K.GT.FINY+1) GOTO 150
                        READ(1)(FZR4(J),J=1,FINX)
90          CONTINUE
C
C START OF THE PIXEL LOOP
      INCX=1
      DO 100  J=INIX,FINX,INTERX
C
C TAKE THE PROPER WINDOW OF 8 BITS
```

```
        IF (WIND.EQ.-1) THEN
                BUFFER = INT((255.0*(FZR4(J)-ZMIN))/(ZMAX-ZMIN))
        ELSE
                Z=INT(FZR4(J))
                CALL MVBITS(Z,WIND,8,BUFFER,0)!TAKE THE PROPER WINDOW IF 8 BITS
        ENDIF
        BUFINT=INT(BUFFER*GAINZ)
C
C EVALUATE THE MAXIMUM FOR THE GIVEN X POSITION
        FACTX=GAINX*INTERX
        FACTY=GAINY*INTERY
        FMAX(INCX)=IMAX0(REGTEMP(INCX),Y0 + BUFINT + INCY*FACTY)
        X = XC + INCX*FACTX + INCY*SHIFT*FACTX
C
C DRAW HORIZONTAL VECTORS IF WANTED
        IF(INCX-1.LT.1) GOTO 95
                IF(AXE.EQ.'DEUX'.OR.AXE.EQ.'deux'
        1       .OR.AXE.EQ.'HOR'.OR.AXE.EQ.'hor') THEN
        DNTCRE=VECTOR(X-FACTX,FMAX(INCX-1),X,FMAX(INCX),IVAL,0)
                ENDIF
C
C DRAW VERTICAL VECTORS IF WANTED
95      IF (INCY.NE.0) THEN
                IF(AXE.EQ.'DEUX'.OR.AXE.EQ.'deux'
        1       .OR.AXE.EQ.'VER'.OR.AXE.EQ.'ver') THEN
        DNTCRE=VECTOR(X-SHIFT*FACTX,FTEMP(INCX),X,FMAX(INCX),IVAL,0)
                ENDIF
        ENDIF
        INCX=INCX+1
100     CONTINUE
                DO 110 J=1,FINX
                FTEMP(J)=FMAX(J)
                IF (J+SHIFT.LT.1) GOTO 110
                REGTEMP(J)=FMAX(J+SHIFT)
110             CONTINUE
        INCY=INCY+1
150     CONTINUE
C
        CLOSE (UNIT=1,ERR=9989)
C
        RETURN
C
9989    CONTINUE
C
        STATUS=RELESE()
        CALL EXIT
C
        RETURN
C
        END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C LECCOOR
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme de mesure entre les coordonnees des deux curseurs
C       graphique. Les mesures sont effectuees avec les facteurs d'echelle
C       et affichees sur le terminal et sur l'ecran graphique.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE LECCOOR (IMA_NAME,INIX,FINX,INIY,FINY)
C
        IMPLICIT        INTEGER*2 (A-Z)
        CHARACTER*30    IMA_NAME
        LOGICAL*4       SUCCESS
C
        INTEGER*4       IOS,KX,NORG,DELX,DELY
        REAL*4          FZR4(512),ZMIN,ZMAX,DX,DY,DZ
        REAL*4          SCL,XOF,YOF,GRDNULL,XINC,YINC
        REAL*4          XMIN,XMAX,YMIN,YMAX
C
C       open image file
C
         OPEN ( UNIT=1, NAME=IMA_NAME, TYPE='OLD', ACCESS='SEQUENTIAL',
        1 FORM='UNFORMATTED',IOSTAT=IOS)
C
C READ THE DIMENSIONS FROM THE STANDARD FILE OR THE PARAMETERS FILE
C
        READ(1) DELX,DELY,SCL,KX,NORG,XOF,YOF,GRDNULL,XINC,YINC
     &  ,XMIN,XMAX,YMIN,YMAX
C
C READ THE LINES OF THE IMAGE WHICH DON'T HAVE TO BE DISPLAY
C
        DO I=1,IMAX0(FINY,INIY)
                IF(I.EQ.INIY.OR.I.EQ.FINY) THEN
                        READ(1)(FZR4(J),J=1,IMAX0(INIX,FINX))
                        IF (I.EQ.INIY) ZMIN=FZR4(INIX)/SCL
                        IF (I.EQ.FINY) ZMAX=FZR4(FINX)/SCL
                ELSE
                        READ(1) FZR4(1)
                ENDIF
        ENDDO
C
        WRITE(5,90) XMIN+INIX*XINC,YMIN+INIY*YINC,ZMIN
        WRITE(5,91) XMIN+FINX*XINC,YMIN+FINY*YINC,ZMAX
        DX = (FINX-INIX)*XINC
        DY = (FINY-INIY)*YINC
        DZ = (ZMAX-ZMIN)
        WRITE(5,100) DX
```

```
      WRITE(5,101) DY
      WRITE(5,102) DZ
      WRITE(5,103) SQRT(DX**2.0+DY**2.0)
      WRITE(5,104) SQRT(DX**2.0+DY**2.0+DZ**2.0)
      CALL AFFICOOR(DX,DY,DZ,IMA_NAME)
C
90          FORMAT(' X1 =',F10.3,' Y1 =',F10.3,' Z1 =',F10.3)
91          FORMAT(' X2 =',F10.3,' Y2 =',F10.3,' Z2 =',F10.3)
100         FORMAT(' Distance X =',F10.3)
101         FORMAT(' Distance Y =',F10.3)
102         FORMAT(' Distance Z =',F10.3)
103         FORMAT(' Distance 2D =',F10.3)
104         FORMAT(' Distance 3D =',F10.3)
C
      CLOSE (UNIT=1,STATUS='KEEP')
C
      RETURN
      END
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C AFFICOOR
C
C Ecrit par : Luc Cournoyer                                  Mai 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Sous-programme qui affiche les coordonnees mesure par LECCOR
C       sur l'ecran graphique.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        SUBROUTINE AFFICOOR(DX,DY,DZ,IMA_NAME)
C
        character*30 ima_name
        logical*1 string(14)
        logical*4 success
C
        SUCCESS = INIT()
        IF ( .NOT. SUCCESS ) CALL FATAL_ERROR( SUCCESS, ' ERROR: IN INIT ')
C
        DIST2D = SQRT(DX**2.0+DY**2.0)
        DIST3D = SQRT(DX**2.0+DY**2.0+DZ**2.0)
C
100     FORMAT(' X =',F8.3)
101     FORMAT(' Y =',F8.3)
102     FORMAT(' Z =',F8.3)
103     FORMAT('2D =',F8.3)
104     FORMAT('3D =',F8.3)
105     FORMAT(1A14)
C
        call alp(5,6,2,56,0,'DISTANCES :')
        encode (14,100,string) DX
        call alp(5,6,4,56,0,string)
        encode (14,101,string) DY
        call alp(5,6,5,56,0,string)
        encode (14,102,string) DZ
        call alp(5,6,6,56,0,string)
        encode (14,103,string) DIST2D
        call alp(5,6,7,56,0,string)
        encode (14,104,string) DIST3D
        call alp(5,6,8,56,0,string)
        encode (14,105,string) IMA_NAME
        call alp(5,6,22,10,0,string)
C
        STATUS = RELESE()
C
        end
```

ANNEXE F

Sources des programmes de transformation et de lecture sur VAX-11/750

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C REDUIT1
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Effectue la reduction de l'image dans les deux axes avec
C       un facteur de reduction X et Y pour chacun d'eux. Le fichier
C       d'entree est de format DPICT.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*2 linumb,pix,fact,factx,facty
        integer*4  xnum, ynum, kx, norg
        real*4 scl, xof, yof, grdnull
        real*4 line(1024),lineout(1024)
30      format(1a80)
C
        type*,'Nom du fichier d''entree (longueur variable) ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read (1) xnum,ynum,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(xi,i=11,126)
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=2,name=outfile,type='new',access='sequential',
     &  form='unformatted')
        type*,'Facteur de reduction (x,y) ?'
        read(5,*) factx,facty
C
C       Ecrit le premier record du fichier de sortie
        write (2) xnum/factx,ynum/facty,scl,kx,norg,
     &  xof,yof,grdnull,xinc,yinc,(i1,j=11,126)
C
C       Depart de la boucle de lecture
        do linumb=1,ynum,facty
                do i=1,facty
                        read (1) (line(j),j=1,xnum)
                enddo
                do pix=factx,xnum,factx
                        lineout(pix/factx) = line(pix)
                enddo
                write (2) (lineout(pix),pix=1,xnum/factx)
        enddo
        close (unit=1,status='keep')
        close (unit=2,status='save')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C EXPAND1
C
C Ecrit par : Luc Cournoyer                                  Fevrier 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Effectue l'expansion de l'image dans les deux axes avec
C       un facteur d'expansion X et Y pour chacun d'eux.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*2 linumb,pix,fact,factx,facty
        integer*4  xnum, ynum, kx, norg
        real*4 scl, xof, yof, grdnull
        real*4 line(1024),lineout(1024)
30      format(1a80)
C
        type*,'Nom du fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read (1) xnum,ynum,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(xi,i=11,126)
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=2,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
        type*,'Facteurs d''expansion (x,y) ?'
        read(5,*) factx,facty
C
C       Ecrit le premier record du fichier de sortie
        write (2) xnum*factx,ynum*facty,scl,kx,norg,
     &  xof,yof,grdnull,xinc,yinc,(i1,j=11,126)
C
C       Depart de la boucle de lecture
        do linumb=1,ynum
                read (1) (line(j),j=1,xnum)
                do pix=1,xnum
                        pixexp = pix*factx - 1
                        do fact=1,factx
                                lineout(pixexp+fact) = line(pix)
                        enddo
                enddo
                do fact=1,facty
                        write (2) (lineout(pix),pix=1,xnum*factx)
                enddo
        enddo
C
```

```
      close (unit=1,status='keep')
      close (unit=2,status='save')
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C MULTIPLIE
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Effectue la multiplication des mots par une valeur donnee
C       par l'usager.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        real*4 multiple,fact,ligne1,ligne2
        integer*2 ligne(512),lineout(512),linumb,pix
30      format(1a80)
C
        type*,'Nom du fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=3,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
        type*,'Facteur de multiplication ?'
        read(5,*) fact
C
        type*,'Dimensions (x,y) ?'
        read(5,*) xnum,ynum
C
C       Depart de la boucle de lecture et de soustraction
        do 1=1,ynum
                read (1) (ligne(i),i=1,xnum)
                do pix=1,xnum
                        multiple = ligne(pix) * fact
                        if (multiple.le.0.0) then
                                lineout(pix) = 0
                        else
                                lineout(pix) = MIN(multiple,32767.0)
                        endif
                enddo
                write (3) (lineout(i),i=1,xnum)
        enddo
C
        close (unit=1,status='keep')
        close (unit=3,status='save')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C OFFSET
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Effectue l'addition d'une valeur fixe aux donnees d'un
C       fichier d'image.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        real*4 addition,offset,ligne1,ligne2
        integer*2 ligne(512),lineout(512),linumb,pix
30      format(1a80)
C
        type*,'Nom du fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=3,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
        type*,'Valeur d''offset ?'
        read(5,*) offset
C
        type*,'Dimensions (x,y) ?'
        read(5,*) xnum,ynum
C
C       Depart de la boucle de lecture et de soustraction
        do 1=1,ynum
                read (1) (ligne(i),i=1,xnum)
                do pix=1,xnum
                        addition = ligne(pix) + offset
                        if (addition.le.0.0) then
                                lineout(pix) = 0
                        else
                                lineout(pix) = addition
                        endif
                enddo
                write (3) (lineout(i),i=1,xnum)
        enddo
C
        close (unit=1,status='keep')
        close (unit=3,status='save')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C JONCTION1
C
C Ecrit par : Luc Cournoyer                                  Septembre 1985
C Division de genie electrique
C Section de photonique et capteurs
C
C              Effectue la jonction de plusieurs images et en forme une
C       nouvelle. On doit specifier le nombre d'images horizontales et
C       verticales. Les images sont de format DPICT.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
      character*80 infile,outfile
      integer*2 imapos,derunit,unite,linumb,pix,posx
      integer*4 kx, norg
      real*4 scl, xof, yof, grdnull, line(512),
     >          min/1000.0/, max, ech
      integer*4 xnum,ynum,numh,numv,numima
30    format(1a80)
C
      type*,'Combien d''images horizontales ?'
      read*,numh
      type*,'Combien d''images verticales ?'
      read*,numv
      numima = numv * numh
C
C     Lecture des noms de fichier d'entree
      do unite=1,numima
            type*,'Nom du',unite,' ier fichier ?'
            read(5,30) infile
            open(unit=unite,name=infile,type='old',access='sequential',
     &  form='unformatted')
            read (unite) xnum,ynum,scl,kx,norg,xof,yof,grdnull,(i1,j=9,126)
            type*,' '
            type*,'Dimensions de l''image #',unite,' (x,y) :',xnum,ynum
            type*,' '
      enddo
C
      type*,' '
      type*,'Nom du fichier de sortie ?'
      read(5,30) outfile
      open(unit=15,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
C     Ecrit le premier record du fichier de sortie
      write (15) xnum*numh,ynum*numv,scl,kx,norg,xof,yof,grdnull,(i1,j=9,126)
C
      derunit = 1
      do imapos=1,numv
            do linumb=1,ynum
                  posx = 1
```

```
                        do unite=derunit,imapos*numh
                                read (unite) (line(pix),pix=posx,posx+xnum-1)
                                posx = posx + xnum
                        enddo
                        write (15) (line(pix),pix=1,numh*xnum)
                enddo
                derunit = imapos*numh + 1
        enddo
C
C       Petite boucle pour fermer les fichiers
        close (unit=15,status='keep')
        do unite=1,numima
                close (unit=unite,status='save')
        enddo
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C SEPAR
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme qui separe les mots 3d et intensite d'une image
C       prise avec la camera 3D rapide.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*2 line(512),lineout(2,512),linumb,pix
30      format(1a80)
C
C
        type*,'Fichier de l''image d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        type*,'Fichier de sortie de l''image 3D ?'
        read(5,30) outfile
        open(unit=2,name=outfile,type='new',access='sequential',
     &  form='unformatted')
        type*,'Fichier de sortie de l''image Intensite ?'
        read(5,30) outfile
        open(unit=3,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
        type*,'Nombre de pixels/ligne ?'
        read(5,*) numpix
        type*,'Nombre de lignes ?'
        read(5,*) numligne
C
C       Depart de la boucle de lecture
        do linumb=1,numligne
                read (1) (line(j),j=1,2*numpix)
                do pix=1,numpix
                   lineout(1,pix) = line(2*pix-1)
                   lineout(2,pix) = line(2*pix)
                enddo
                write (2) (lineout(1,pix),pix=1,numpix)
                write (3) (lineout(2,pix),pix=1,numpix)
        enddo
C
        close (unit=1,status='keep')
        close (unit=2,status='save')
        close (unit=3,status='save')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C TRANSVAR
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Convertit une image de longueur de record fixe a une image
C       de longueur de record variable. On doit specifier la dimension de
C       l'image.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*2 ima(65536),linumb
        real*4 numrec,xnum,ynum,rec
30      format(1a80)
C
        rec=512
C
        type*,'Nom du fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=2,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
        type*,'Dimensions (x,y) ?'
        read(5,*) xnum,ynum
C
C       Depart de la boucle de lecture
        icount=0
        pixmin=1
        pixmax=rec
        numrec=xnum*ynum
        numrec=MIN0(128,INT(numrec/rec)+1)
        type*,'Nombre de record de 512 mots :',numrec
        do linumb=1,numrec
                read (1) (ima(j),j=pixmin,pixmax)
                if(ima(pixmax).eq.-1) then
                        ima(pixmax)=ima(pixmax-1)
                        icount=icount+1
                endif
                pixmin=pixmin+rec
                pixmax=pixmax+rec
        enddo
C
        pixmin=1
        pixmax=xnum
```

```
      do linumb=1,ynum
              write (2) (ima(k),k=pixmin,pixmax)
              pixmin=pixmin+xnum
              pixmax=pixmax+xnum
      enddo
C
999   format ('0','Detection de',i3' fin(s) de bloc.')
      write(6,999) icount
C
      close (unit=1,status='keep')
      close (unit=2,status='save')
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C MOYLIGNE1
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Effectue la moyenne des M lignes d'une image de format DPICT
C       et ecrit le resultat dans un nouveau fichier.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        real*4 linemoy(512)/512*0.0/
        integer*4  xnum, ynum, kx, norg
        real*4 scl, xof, yof, grdnull, line(512)
30      format(1a80)
C
        type*,'Nom du fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read (1) xnum,ynum,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(xi,i=11,126)
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=2,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
C       Depart de la boucle de lecture
        do j=1,ynum
                read (1) (line(i),i=1,xnum)
                do k=1,xnum
                        linemoy(k)=linemoy(k)+line(k)
                        if(j.eq.ynum) line(k)=linemoy(k)/ynum
                enddo
        enddo
C
C
C       Ecrit le premier record du fichier de sortie
        ynum = 1.0
        write (2) xnum,ynum,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(xi,i=11,126)
C
        write (2) (line(i),i=1,xnum)
C
        close (unit=1,status='keep')
        close (unit=2,status='save')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C SOUSTR1
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de soustraction de deux images de format DPICT de
C       meme dimensions.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile,outfile
        integer*4  xnum, ynum, kx, norg
        real*4 soustrait,offset,ligne1,ligne2
        real*4 ligne(512),ligneref(512),lineout(512),linumb,pix
        real*4 scl, xof, yof, grdnull
30      format(1a80)
C
        type*,'Nom du premier fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read (1) xnum,ynum,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(xi,i=11,126)
C
        type*,'Nom du fichier a soustraire du premier ?'
        read(5,30) infile
        open(unit=2,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read (2) xnum,ynum,scl,kx,norg,xof,yof,grdnull,
     &  xinc,yinc,(xi,i=11,126)
C
        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=3,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
C       Ecrit le premier record du fichier de sortie
        write (3) xnum,ynum,scl,kx,norg,xof,yof,
     &  grdnull,xinc,yinc,(i1,j=11,126)
C
        type*,'Valeur d''offset ?'
        read(5,*) offset
C
C       Depart de la boucle de lecture et de soustraction
        do i=1,ynum
                read (1) (ligne(i),i=1,xnum)
                read (2) (ligneref(i),i=1,xnum)
                do pix=1,xnum
                        soustrait = ligne(pix) - ligneref(pix) + offset
                        if (soustrait.le.0.0) then
```

```
                        lineout(pix) = 0
                    else
                        lineout(pix) = soustrait
                    endif
                enddo
                write (3) (lineout(i),i=1,xnum)
        enddo
C
        close (unit=1,status='keep')
        close (unit=2,status='keep')
        close (unit=3,status='save')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C PIVOTJ
C
C Ecrit par : Luc Cournoyer                                   Juin 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C               Programme de rotation d'une image autour d'un axe qui
C       peut etre :
C
C               a)  vertical et place au centre de l'image originale
C               b)  horizontal et place au centre de l'image originale
C               c)  normal au plan de l'image originale
C
C       Le fichier d'entree doit etre de format DPICT
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*1 rotype
        character*80 infile,outfile
        real*4 line(1024,1024),lineout(1024)
20      format(1a1)
30      format(1a80)
C
        type*,'Nom du fichier d''entree ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')

        type*,'Nom du fichier de sortie ?'
        read(5,30) outfile
        open(unit=2,name=outfile,type='new',access='sequential',
     &  form='unformatted')
C
        read (1) numpix,numligne,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,(i1,i=11,126)
        type*,numpix,numligne
C
        type*,' '
        type*,'Quelle transformation ?'
        type*,' '
        type*,'          A - Rotation autour de l''axe vertical'
        type*,'          B - Rotation autour de l''axe horizontal'
        type*,'          C - Rotation autour de la normale au plan'
        type*,'          D - Aucune rotation '
        type*,' '
        read(5,20) rotype
        type*,' '
C
        if (rotype.eq.'d'.or.rotype.eq.'D') then
                stop
        endif
```

```
C
      if (rotype.eq.'A'.or.rotype.eq.'a') then
            write (2) numpix,numligne,scl,kx,norg,xof,yof,grdnull
     &      ,xinc,yinc,(i1,i=11,126)
            do linumb=1,numligne
                  read (1) (line(1,j),j=1,numpix)
                  do pix=1,numpix
                        lineout(pix) = line(1,numpix-pix+1)
                  enddo
                  write (2) (lineout(pix),pix=1,numpix)
            enddo

      else
            if (rotype.eq.'B'.or.rotype.eq.'b'
     & .or.rotype.eq.'C'.or.rotype.eq.'c') then
C     Depart de la boucle de lecture du fichier d'entree
                  do linumb=1,numligne
                        read (1) (line(linumb,j),j=1,numpix)
                  enddo

      if (rotype.eq.'b'.or.rotype.eq.'B') then
C     Depart de la boucle de rotation horizontale
                  write (2) numpix,numligne,scl,kx,norg,xof,yof,grdnull
     &            ,xinc,yinc,(i1,i=11,126)
                  do linumb=numligne,1,-1
                        write (2) (line(linumb,pix),pix=1,numpix)
                  enddo

      else
C     Depart de la boucle de rotation sur l'axe des Z
                  write (2) numligne,numpix,scl,kx,norg,xof,yof,grdnull
     &            ,yinc,xinc,(i1,i=11,126)
                  do linumb=1,numpix
                        write (2) (line(pix,linumb),pix=1,numligne)
                  enddo
      endif
      endif
      endif
C
      close (unit=1,status='keep')
      close (unit=2,status='save')
C
      end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C LECTURE
C
C Ecrit par : Luc Cournoyer
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de lecture des valeurs d'une image provenant directement
C       du capteur 3D.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile
        integer*2 ligne(512),linumb,pix
30      format(1a80)
50      format('      pixel #',I3,' = ',I8)
C
        type*,'Nom du fichier a lire ?'
        read(5,30) infile
C
100     open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        type*,'Combien de pixels a afficher ?'
        read(5,*) xnum
        type*,'Quelle ligne est a afficher ?'
        read(5,*) lignelec
C
C       Depart de la boucle de lecture
        do 1=1,lignelec
                read (1) (ligne(i),i=1,xnum)
        enddo

        type*,'ligne #',1-1
        type*,' '
        type*,' '

        do pix=1,xnum
                write(*,50),pix,ligne(pix)
        enddo
C
        close (unit=1,status='keep')
C
        type*,'Encore (1=oui) ?'
        read*,boucle
        if (boucle.eq.1) goto 100
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C LECTURE1
C
C Ecrit par : Luc Cournoyer                                  Novembre 1985
C Division de genie electrique
C Section de photonique et capteurs
C
C       Programme de lecture des valeurs d'une image de format DPICT.
C       La sortie peut etre corrigee ou non par le facteur d'echelle
C       contenue dans le premier registre de l'image.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile
        real*4 ligne(1024)
        integer*4 dimx,dimy,linumb,pix
30      format(1a80)
50      format(' Ligne #',I4,'      , pixel #',I4,' = ',F15.6)
C
        type*,'Nom du fichier a lire ?'
        read(5,30) infile
C
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read(1) dimx,dimy,scl,kx,norg,xof,yof,grdnull,xinc,yinc
        type*,' '
        type*,'Dimensions (x,y) : ',dimx,dimy
        type*,'Facteur d''echelle "X" : ',xinc
        type*,'Facteur d''echelle "Y" : ',yinc
        type*,'Facteur d''echelle "Z" : ',scl
        type*,' '
        type*,'Lecture sur l''axe des X (1=oui) ?'
        read*,axe
C
100     rewind (1)
        read(1) dimx,dimy,scl,kx,norg,xof,yof,grdnull,xinc,yinc
        type*,'Nombre de pixels a afficher ?'
        read(5,*) xnum
        if (axe.eq.1) then
                type*,'Quelle ligne est a afficher ?'
                read(5,*) ydep
                type*,'A partir de quelle colonne ?'
                read(5,*) xdep
                ylec = 0
                xlec = xnum
        else
                type*,'Quelle colonne est a afficher ?'
                read(5,*) xdep
                type*,'A partir de quelle ligne ?'
                read(5,*) ydep
                xlec = 0
                ylec = xnum
```

```
        endif
        type*,'Presentation corrigee avec le facteur d''echelle (1=oui) ?'
        read*, cor
C
C Lecture des lignes qui n'ont pas a etre affiches
        do l=1,ydep-1
                read (1) ligne(1)
        enddo
C
        indice=1
        do i=ydep,ydep+ylec
        read (1) (ligne(pix),pix=1,xdep+xlec)
        if (cor.eq.1) then
C Presentation corrigee
                do pix=xdep,xdep+xlec
                        write(*,50),i,pix,ligne(pix)/scl
C                       ,200+indice*50
                        indice=indice*-1
                enddo
        else
C Presentation non corrigee
                do pix=xdep,xdep+xlec
                        write(*,50),i,pix,ligne(pix)
C                       ,200+indice*50
                        indice=indice*-1
                enddo
        endif
        enddo
C
        type*,'Encore (1=oui) ?'
        read*,boucle
        if (boucle.eq.1) goto 100
C
        close (unit=1,status='keep')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C LECTURE2
C
C Ecrit par : Luc Cournoyer                                  Avril 1985
C Division de genie electrique
C Section de photonique et capteurs
C
C       Lit les valeurs d'une image pour la ligne specifiee
C       (Format DPICT BINAIRE, XYZ)
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile
        real*4 pixnum,pixmax,depart,count
        dimension x(300000),y(300000),z(300000)
C
30      format(1a80)
40      format('0','Pixel #           X             Y             Z')
50      format(F8.0,':',3F15.6)
C
        type*,'Nom du fichier a lire ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
C
        type*,'Combien de pixels a lire ?'
        read*,pixmax
        do 1=1,pixmax
                read (1) x(1),y(1),z(1)
        enddo
C
99      type*,'Combien de pixels a afficher ?'
        read*,pixnum
100     rewind (1)
        type*,' '
        type*,'A partir de quel pixel (-1=stop,-2=changer pixnum) ?'
        read(5,*) depart
        if (depart.eq.-1) goto 200
        if (depart.eq.-2) goto 99
C
        write(*,40)
        do v=depart,depart+pixnum-1
                write(*,50) v,x(v),y(v),z(v)
        enddo
C
        goto 100
C
200     close (unit=1,status='keep')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches        National Research Council
C Canada                                Canada
C
C LECTURE3
C
C Ecrit par : Luc Cournoyer                             Avril 1985
C Division de genie electrique
C Section de photonique et capteurs
C
C       Lit les valeurs d'une image pour la ligne specifiee
C       (Format DPICT BINAIRE)
C       Lecture dans l'axe des Y, avec interpolation pour plus
C       de precision.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile
        real*4 pixnum,pixmax
C
 30     format(1a80)
 40     format('0','Pixel #          X              Y              Z')
 50     format(F8.0,':',3F15.6)
C
        type*,'Nom du fichier a lire ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        type*,'Combien de pixels a lire ?'
        read*,pixnum
 100    rewind (1)
        type*,' '
        type*,'Bornes minimum en X (-1=stop) ?'
        read(5,*) xmin
        if (xmin.eq.-1) goto 200
C
        write(*,40)
        ref = 1
        xr = 0
        do v=1,pixnum
                read (1) x,y,z
                if (x.lt.xr) ref=1
                if (x.ge.xmin.and.ref.eq.1) then
                        slope = (z-zr)/(x-xr)
                        zout = slope*xmin + z - slope*x
                        write(*,50) v,xmin,y,zout
                        ref = 0
                endif
                xr = x
                zr = z
        enddo
        goto 100
 200    close (unit=1,status='keep')
C
        end
```

```
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C Conseil National de Recherches          National Research Council
C Canada                                  Canada
C
C LECENT
C
C Ecrit par : Luc Cournoyer                                  Avril 1986
C Division de genie electrique
C Section de photonique et capteurs
C
C       Lit l'entete des fichiers de format DPICT pour connaitre
C       les parametres contenues dans le premier record.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
        character*80 infile
        real*4 ligne(1024)
        integer dimx,dimy,linumb,pix
        logical sortie
30      format(1a80)
260     format ('0','Le facteur d''echelle en Z est de ',F10.4)
270     format ('0','La dimension des pixels en X est de ',
     &  F10.5,' unite(s)')
280     format (' L''affichage en X se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
290     format ('0','La dimension des pixels en Y est de ',
     &  F10.5,' unite(s)')
300     format (' L''affichage en Y se fait de ',F10.5,
     &  ' a ',F10.5,' unites')
C
        sortie = 1
        do while (sortie)
        type*,'Nom du fichier a lire ?'
        read(5,30) infile
        open(unit=1,name=infile,type='old',access='sequential',
     &  form='unformatted')
        read (1) dimx,dimy,scl,kx,norg,xof,yof,grdnull
     &  ,xinc,yinc,xmin,xmax,ymin,ymax,(i1,i=15,126)
        type*,' '
        type*,'Dimensions (x,y) : ',dimx,dimy
        write(6,260) scl
        write(6,270) xinc
        write(6,280) xmin,xmax
        write(6,290) yinc
        write(6,300) ymin,ymax
        type*,' '
        type*,'Un autre fichier (1=oui) ?'
        read*,sortie
        close (unit=1,status='keep')
        enddo
C
        end
```